

# Contents

<b>1 引言</b>	<b>3</b>
1.1 考核方式及成绩构成 . . . . .	3
1.2 大作业 . . . . .	3
1.2.1 内容与形式 . . . . .	3
1.2.2 格式 . . . . .	4
<b>2 线性规划</b>	<b>5</b>
2.1 线性规划问题 . . . . .	5
2.1.1 线性规划问题的定义 . . . . .	5
2.1.2 线性规划问题的图解法求解 . . . . .	5
2.2 线性规划问题的数学模型 . . . . .	8
2.3 线性规划解的基本概念与基本理论 . . . . .	10
2.3.1 解 . . . . .	10
2.3.2 基 . . . . .	10
2.3.3 基解 . . . . .	10
2.3.4 线性规划的几个重要定理 . . . . .	12
2.4 线性规划求解的单纯形法 . . . . .	14
2.4.1 初始基可行解的确定 . . . . .	14
2.4.2 寻找另一个基可行解：基变换法 . . . . .	16
2.4.3 最优性检验方法 . . . . .	17
2.5 线性规划问题的灵敏度分析 . . . . .	22
2.6 应用案例分析 . . . . .	22
2.6.1 下料问题 . . . . .	22
2.6.2 连续投资问题 . . . . .	25
2.7 第二章作业 . . . . .	27
2.7.1 配餐问题 . . . . .	27
2.7.2 战略轰炸问题 . . . . .	33
<b>3 运输规划</b>	<b>36</b>
3.1 运输问题 . . . . .	36
3.1.1 产销平衡 . . . . .	36

---

3.1.2 产销不平衡 . . . . .	38
<b>4 整数规划</b>	<b>41</b>
4.1 整数规划问题与数学模型 . . . . .	41
4.1.1 整数规划问题的定义 . . . . .	41
4.1.2 整数规划问题的数学模型 . . . . .	41
4.2 一般整数规划的求解方法—分枝定界法 . . . . .	43
4.3 0-1 规划及其求解方法 . . . . .	51
4.3.1 0-1 规划的定义和数学模型 . . . . .	51
4.3.2 0-1 规划的例题 . . . . .	52
4.3.3 0-1 规划的求解 . . . . .	54
4.4 案例分析 . . . . .	56
4.5 第四章作业 . . . . .	59
4.5.1 车间生产问题（分枝定界法） . . . . .	59
4.5.2 旅行者背包问题（0-1 规划） . . . . .	63
<b>5 非线性规划</b>	<b>69</b>
5.1 非线性规划问题与数学模型 . . . . .	69
5.1.1 非线性规划问题的定义 . . . . .	69
5.1.2 非线性规划问题的数学模型 . . . . .	71
5.1.3 求解非线性规划问题的解析法 . . . . .	73
5.1.4 求解非线性规划问题的数值法 . . . . .	75
5.2 无约束非线性规划问题的解 . . . . .	79
5.3 约束非线性规划问题（约束极值问题） . . . . .	79
5.4 案例分析 . . . . .	79

# 引言

## 1.1 考核方式及成绩构成

本处说明在可预见的时间范围内仅适用于梁军老师班级的同学。

- 本课程没有期末大考;
- 本课程的成绩构成为: 平时成绩 + 期末成绩, 其中期末成绩占比 60%;
- 平时成绩包括:
  - 课外练习, 主要涉及运筹学的基本概念理论、尤其算法, 有一些编程工作;
  - 课堂讨论;
- 期末成绩包括:
  - 课程结束时有一个 30 分钟左右随堂小考 (闭卷), 主要涉及运筹学的基本概念、方法;
  - 期末大作业含大作业的研究报告和结题答辩;

## 1.2 大作业

### 1.2.1 内容与形式

1. 大作业以小组为单位(每个小组 4-6 人, 1 位组长, 人员自由组阁); 要求第二周就要分好组(定下组员及组长), 第三周就要初步定下题目(以后可以改), 并开始调研工作;
2. 为帮助大家定题和开展工作, 第三周结束前由老师抽一点时间与大家交流一下; 另外, 近几年的范例会发给大家;
3. 每个小组就自选的某个社会问题或工程问题, 结合课程所学知识和方法进行分析、建模、优化、预测、评价、决策, 最后给出解决问题的定性、定量化建议或解决方案, 并撰写成科技报告或科技论文(格式见后);

4. 将大作业论文整理成 ppt, 进行课堂现场宣讲和答辩, 由任课教师、助教和其他同学作为评委和质询者, 教师、助教和同学们当堂评分(同学们的分数以小组名义给出, 每个小组给一个分数), 所以大作业答辩课小组成员尽量坐在一起;
5. 答辩过程中小组同学拍照留念(集体照、演讲照、回答问题交流照), 代表性照片插入到研究报告中;
6. 答辩后各小组根据情况(老师和同学们的建议、存在的问题未尽的内容等等), 进行修改、定稿;
7. 关于讨论课和大作业完成后所提交的材料:word+ppt+pdf 文件是必须的, 如果在准备中用到或形成了一些电子材料(如图片、视频、程序、软件、下载的资料、其他), 也一并上交(显示了同学们在讨论课环节上的深入和广泛程度, 展现了工作量), 加分更多;

### 1.2.2 格式

1. 题目、人员及分工、指导老师(2个老师都写)、开始时间、结束时间;
2. 中文摘要、关键词, 英文摘要、关键词;
3. 正文;
  - (a) 引言或问题的提出或需要解决的问题或研究目的等等-说清楚做什么;
  - (b) 理论方法方面的描述、讨论、引用等等-说清楚用什么方法做和怎么做的;
  - (c) 具体应用或解决方案或实验结果等等-做的结果如何;
  - (d) 分析、讨论和结论-得到了什么;
4. 参考文献;
5. 附录、附件(一些电子材料, 如图片、视频、程序、软件、下载的资料、其他);

# 线性规划

## 2.1 线性规划问题

### 2.1.1 线性规划问题的定义

线性规划问题可以归结为求目标函数在约束条件下的最大值问题。

线性规划模型由以下三个基本要素构成：

- **决策变量：**决策变量是问题中要确定的未知量，决策者通过调控决策变量来选取不同的方案、设计、措施以达到最优目的。
- **目标函数：**目标函数通常是决策变量的函数，表达了“何为最优”的准则和目标，规定了优化问题的实际意义。
- **约束条件：**约束条件指决策变量取值时受到的各种资源和条件的限制，表达了一种“有条件优化”的概念，通常为决策变量的等式或不等式方程。

#### Definition 2.1.1 ▶ 线性规划问题

线性规划问题是一类决策变量的取值是连续的，且目标函数和约束条件都是决策变量的线性函数的问题

- **整数规划问题：**决策变量的取值为整数点。
- **混合整数规划问题：**部分决策变量取值连续而其余取值为整数。
- **非线性规划问题：**目标函数或约束条件中存在任何的非线性因子。

### 2.1.2 线性规划问题的图解法求解

目前，线性规划问题的求解方法主要有两种：

- **图解法：**适用于只有两个决策变量的线性规划问题。其可行域可以在平面上画出。
- **单纯形法：**适用于三个决策变量以上数决策变量的线性规划问题。

#### Definition 2.1.2 ▶ 解的可行域

解的可行域是满足约束条件的决策变量向量在  $n$  维空间中构成的点的集合。

可行域中使得目标函数达到最优的解点成为**最优解**，相应的目标函数值称为**最优值**。  
求解步骤如下：

1. 建系：以两个**决策变量**为轴在平面上建立直角坐标系
2. 可行域：由线性等式和不等式构成的**约束条件**，标出可行域
3. 最优解：图示并移动**目标函数**，寻找最优解。

### Example 2.1.3 ▶ 图解法解线性规划

用图解法解下列线性规划：

$$\begin{aligned} & \min -x_1 - 4x_2 \\ \text{s.t. } & x_1 + x_2 \leq 4 \\ & -x_1 + x_2 \leq 2 \\ & x_1, x_2 \geq 0 \end{aligned}$$

解：

1. 以  $x_1, x_2$  为坐标轴画出直角坐标系；
2. 分别画出  $x_1 + x_2 = 4, -x_1 + x_2 = 2, x_1 = 0, x_2 = 0$  四条直线，则该问题的可行域为这四条直线包围的内部区域  $S$ ；
3. 目标函数的等值线方程为  $-x_1 - 4x_2 = z$ 。因为要找的最优解在可行域内使目标函数具有最小值，所以让等值线  $-x_1 - 4x_2 = z$  沿  $z$  减小的方向在可行域内尽量平行移动，直到图中  $x_1 = 1, x_2 = 3$  的位置，如果再移动就移出了可行域  $S$ 。于是，点  $(1,3)$  即为问题的最优解，目标函数的最优值为  $-13$ 。

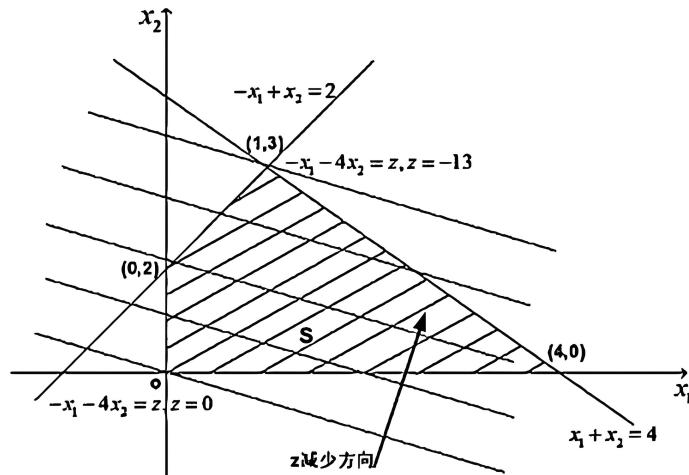
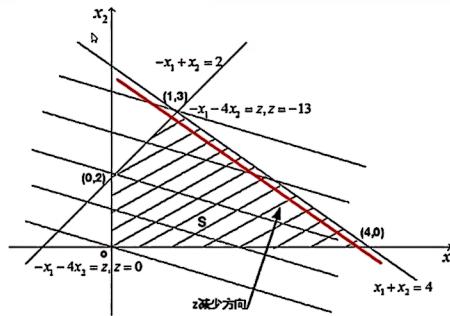


Figure 2.1: 例 1.1.3 图解

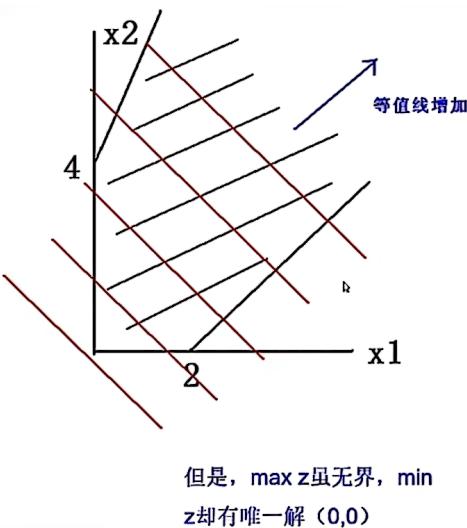
思考:

- **无穷多解:** 上例中若取  $\min z = 3x_1 + 3x_2$ , 则目标函数线与  $x_2 + x_2 = 4$  边界线重合, 带来无穷多解。



- **无界解:** 例如下述线性规划问题:

$$\begin{aligned} \max z &= x_1 + x_2 \\ \text{s.t.} \quad &-2x_1 + x_2 \leq 4 \\ &x_1 - x_2 \leq 2 \\ &x_1, x_2 \geq 0 \end{aligned}$$



- 无最优解: 若可行域为空, 则无可行解, 自然无最优解。

## 2.2 线性规划问题的数学模型

### Theorem 2.2.1 ▶ 一般形式

特点: 目标函数和约束条件都是决策变量的线性函数。

$$\begin{aligned} \max(\min) z &= \sum_{j=1}^n c_j x_j \\ \text{s.t. } &\left\{ \begin{array}{l} \sum_{j=1}^n a_{ij} x_j \leq (\geq, =) b_i \quad (i = 1, 2, \dots, m), \\ x_j \geq 0 \quad (j = 1, 2, \dots, n). \end{array} \right. \end{aligned}$$

也可以表示为矩阵形式:

$$\begin{aligned} \max(\min) z &= \mathbf{c} \cdot \mathbf{x} \\ \text{s.t. } &\left\{ \begin{array}{l} \mathbf{Ax} \leq (\geq, =) \mathbf{b}, \\ \mathbf{x} \geq 0 \end{array} \right. \end{aligned}$$

其中称  $\mathbf{c} = (c_1, c_2, \dots, c_n)$  为目标函数的系数向量;

称  $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$  为决策向量;

称  $\mathbf{b} = (b_1, b_2, \dots, b_m)$  为约束方程组的常数向量;

称  $\mathbf{A} = (a_{ij})_{m \times n}$  为约束方程组的系数矩阵;

称  $p_j = (a_{1j}, a_{2j}, \dots, a_{mj})^T$  为约束方程组的系数向量.

**Theorem 2.2.2 ▶ 标准形**

为了便于研究，规定线性规划模型的标准型。

$$\begin{aligned} \max z &= \mathbf{c} \cdot \mathbf{x} \\ \text{s.t. } &\begin{cases} \mathbf{A}\mathbf{x} = \mathbf{b}, \\ \mathbf{x} \geq 0 \end{cases} \end{aligned}$$

**方法：一般形式化为标准形的方法**

- 目标函数：**若目标函数为  $\min z$ ，则令  $z' = -z$  转化为  $\max z'$ 。
- 约束条件：**若约束条件为不等式，可以再不等号左端加上/减去一个非负变量（称为松弛变量）化为等式约束（哪边小，加哪边）。
- 决策变量：**若决策变量非正 ( $x_j \leq 0$ )，则令  $x'_j = -x_j$ ,  $x'_j \geq 0$ ，转化为非负变量。

**Example 2.2.3 ▶ 一般形式化标准形**

$$\begin{aligned} \min z &= x_1 + 2x_2 - 3x_3 \\ x_1 + x_2 + x_3 &\leq 7 \\ x_1 + x_2 + x_3 &\geq 2 \\ -3x_1 + x_2 + 2x_3 &= 5 \\ x_1, x_2 &\geq 0 \end{aligned}$$

化为标准形式。

解：

- 因  $x_3$  无约束，令  $x_3 = x_4 - x_5$ ,  $x_4, x_5 \geq 0$
- $x_1 + x_2 + x_3 \leq 7 \rightarrow x_1 + x_2 + (x_4 - x_5) + \cancel{x_6} = 7$
- $x_1 + x_2 + x_3 \geq 2 \rightarrow x_1 + x_2 + (x_4 - x_5) = 2 + \cancel{x_7}$
- $-3x_1 + x_2 + 2x_3 = 5 \rightarrow -3x_1 + x_2 + 2(x_4 - x_5) = 5$

其中： $x_1, x_2, x_4, x_5, x_6, x_7 \geq 0$

此时再考虑目标函数

$$\min z = -x_1 + 2x_2 - 3x_3 \rightarrow \max z' = -z = x_1 - 2x_2 + 3x_4 - 3x_5 + 0x_6 + 0x_7$$

## 2.3 线性规划解的基本概念与基本理论

### 2.3.1 解

#### Definition 2.3.1 ▶ 可行解

满足 Theorem 2.2.2 约束条件的解  $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$  为线性规划问题的可行解。

#### Definition 2.3.2 ▶ 可行域

可行解全体构成的集合称为可行域，记为  $D$ 。

#### Definition 2.3.3 ▶ 最优解

使 Theorem 2.2.2 中目标函数达到最大的可行解称为最优解。

### 2.3.2 基

#### Definition 2.3.4 ▶ 基

设系数矩阵  $A = (a_{ij})_{m \times n}$  的秩为  $m$ ，则称  $A$  的某个  $m \times m$  非奇异子矩阵  $B$  为线性规划问题的一个基。

不妨设  $B = (a_y)_{m \times m} = (\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m)$

- **基向量：** 向量  $\mathbf{p}_j = (a_{1j}, a_{2j}, \dots, a_{mj})^T$  ( $j = 1, 2, \dots, m$ );
- **非基向量：** 矩阵  $A$  的其他列向量;
- **基变量：** 与基向量对应的决策变量  $x_j$  ( $j = 1, 2, \dots, m$ );
- **非基变量：** 其他的变量称为非基变量。

### 2.3.3 基解

设问题的基为  $B = (\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m)$ ，将约束为：

$$\sum_{j=1}^m \mathbf{p}_j x_j = \mathbf{b} - \sum_{j=m+1}^n \mathbf{p}_j x_j \quad (2.1)$$

**Definition 2.3.5 ▶ 基解**

在方程组 (2.1) 的解中, 令  $x_j = 0(j = m + 1, m + 2, \dots, n)$ , 则解向量  $\mathbf{x} = (x_1, x_2, \dots, x_m, \dots, 0, 0)$  为问题的基解。

**Definition 2.3.6 ▶ 基可行解、可行基**

满足非负约束条件的基解称为基可行解, 对应于基可行解的基解为可行基。

**方法:** 理解上述定义 对于  $A\mathbf{x} = \mathbf{b}$  这样一个矩阵方程, 我们以下面为例

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{37} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} & a_{57} \end{bmatrix} \quad \text{and} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}$$

观察, 不难发现以下几个性质:

- $A$  是一个  $m \times n$  的矩阵, 其中  $m < n$ , 这样决策变量才有多解; 如果  $m = n$  即满秩, 就只有唯一解了。
- 在矩阵  $A$  中选取一个  $m \times m$  的子矩阵  $B$ , 并且这个子矩阵是非奇异的, 那么就可以得到一个基, 如标红所示。该子矩阵的  $m$  个行向量线性无关,  $m$  个列向量也线性无关。因此相当于一个  $m$  维空间中的坐标系。
- 基  $B$  中的每一个列向量就是基向量, 不属于  $B$  但在  $A$  中的列向量就是非基向量。
- $A$  右乘列向量  $\mathbf{x}$ ,  $\mathbf{x}$  中标红的变量就是与基向量对应的决策变量, 其他未标红的变量就是非基变量。
- 把基变量留下来, 把非基变量移到等式右侧, 令非基变量为 0, 得到的解就是基解; 也就是将基  $B$  中的列向量与  $\mathbf{x}$  中的基变量相乘, 得到的就是基解。换句话说, 令  $\mathbf{x}$  中的非基变量为 0, 左乘  $A$  就可以得到基解;
- 如果基解本身均大于 0, 就是基可行解。

**Definition 2.3.7 ▶ 凸集**

假设  $K$  为  $n$  维欧氏空间中的点集，如果对于任意两点，其连线上所有点均在  $K$  内，则称  $K$  为凸集。

例：实心圆、实心球是凸集，空心圆、空心球不是凸集。直观地说，凸集没有凹入部分。

**Definition 2.3.8 ▶ 凸集的顶点**

对于凸集  $K$  中的点  $x$ ，如果  $x$  不能用相异的两点  $x^{(1)}, x^{(2)} \in K$  的凸组合表示为  $x = \lambda x^{(1)} + (1 - \lambda)x^{(2)} \in K$  ( $0 \leq \lambda \leq 1$ )，即点  $x$  不在  $x^{(1)}$  和  $x^{(2)}$  的连线上。则称  $x$  为凸集  $K$  的一个顶点

对于凸集，顶点一般为边界点，但并非所有边界点都是顶点。

### 2.3.4 线性规划的几个重要定理

基于 2.3 节中代数学中求解方程和集合论的铺垫之后，我们可以得到几个  $z$  重要定理：

**Theorem 2.3.9 ▶ 定理**

1. 如果线性规划问题存在可行域  $D$ ，则其可行域  $D = \{x \mid \sum_{j=1}^n p_j x_j = b, x_j \geq 0\}$  一定是凸集。
2. 线性规划问题 (2.5), (2.6) 的任一个基可行解  $x$  必对应于可行域  $D$  的一个顶点。
3. 对于可行域
  - 如果可行域有界，则问题的最优解一定在可行域的顶点上达到。
  - 如果可行域无界，则问题可能无最优解；若有最优解也一定在可行域的某个顶点上达到。

定理 1 的好处在于，凸集有很多良好的性质。

定理 2 对我们求解没有太大帮助。

定理 3 将求解最优化的可行域中无穷可解点的比较问题变成了三个定点的比较问题。

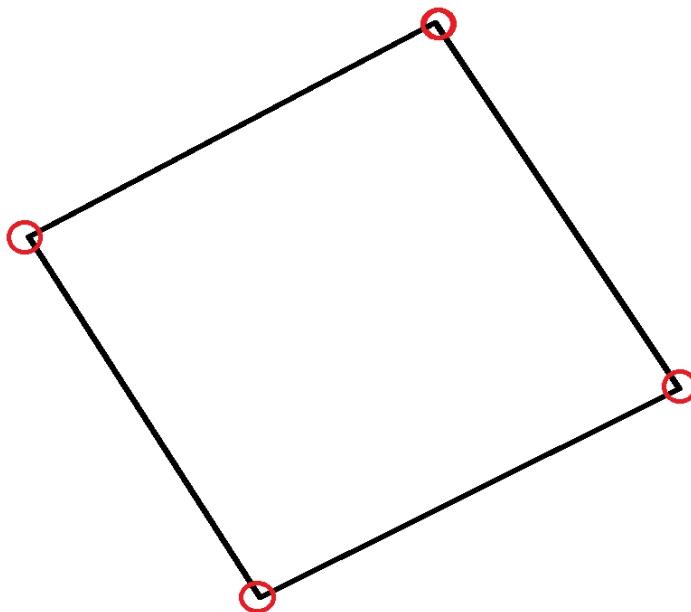


Figure 2.2: 例如我们的可行域是上图中的这个四边形，最优解一定在这个四边形的四个顶点上，其他的可行域的点全都不必算。这样我们把无穷维（有无穷个点）的优化问题转化为有限的、可穷举的优化问题。

总结来说，可以得到四个结论：

1. 线性规划所有可行解构成的集合为**凸集**，也可能是**无界域**；
2. 线性规划的可行域有**有限**个顶点；
3. 线性规划的每个基可行解对应可行域的**一个顶点**；
4. 若线性规划有最优解，必定在**某个顶点**上达到，但**并非只能在顶点上达到**。  
(比如一个函数在某处有最大值，不是说只有在这一处能达到最大值，可能  $f(x_1) = f(x_2) = f_{max}, x_1 \neq x_2$ )

本质上可以理解为一个可行域内的点，只有顶点是线性无关的，其他任何点都可以由顶点线性表示。

这些结论构成了单纯形法的理论基础。

## 2.4 线性规划求解的单纯形法

求解线性规划问题的方法

1. 求一个基可行解(即对应可行域的一个顶点);
2. 检查该基可行解是否为最优解;
  - 如果不是, 则设法再求另一个没有检查过的基可行解(可行域内另一个顶点), 如此进行下去, 直到得到某一个基可行解为最优解为止;
  - 如果是, 结束。

这个过程本质上就是先找到可行域内任意一个顶点, 然后跳到其他顶点上, 穷举的办法求谁最大。就好像我们在函数中, 有许多极大值, 但是要都求出来去求最大值。如果用函数来类比, 2.4.1中所做的就是先找到任意一个极大值点  $x_1$ , 2.4.2中所做的就是利用  $x_1$  便捷地去找其他极大值点;

那么我们不禁要问:

- 如何求出第一个基可行解? (如何找到  $x_1$ ?)
- 如何由一个基可行解过渡到另一个基可行解? (怎么通过  $x_1$  快速找到其他极大值点?)
- 如何判断基可行解是否为最优解? (哪个极大值点是最大值点?)

解决这些问题的方法称为单纯形法。

### 2.4.1 初始基可行解的确定

初始基可行解的确定方法

1. 标准化: 第一步, 将线性规划模型化为标准型;
2. 解方程: 第二步, 解矩阵方程, 得到基可行解。

这一步骤本质上是找到可行域的任意一个顶点, 且只能硬着头皮算。考虑到读者可能和笔者一样忘记了线性代数的相关内容, 因此在这里对矩阵方程解法加以补充: 现在假设我们有:

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ s_1 \\ s_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 4 \\ 5 \\ 7 \end{bmatrix}$$

- 从系数矩阵  $A = (a_{ij})_{m \times n}$  ( $m < n$ , 秩为  $m$ ) 总可以得到一个  $m$  阶单位阵  $E_m$ 。(例如, 可以通过高斯消元法对系数矩阵  $A$  进行行初等变换, 得到一个  $m$  阶单位阵  $E_m$  )。  
 $Ax = b \rightarrow [E_m \text{ 其他元素}] x' = b'$ .

$$A \xrightarrow{\text{初等变换}} \begin{bmatrix} 1 & 0 & 0 & -2 & -3 \\ 0 & 1 & 0 & -3 & -4 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

- 取如上  $m$  阶单位阵  $E_m$  为初始可行基, 即  $B = E_m$ , 将相应的约束方程组变为:  
 $x_i = b_i - a_{i,m+1}x_{m+1} - \dots - a_{i,n}x_n \quad (i = 1, 2, \dots, m)$ .

$$\begin{bmatrix} 1 & 0 & 0 & -2 & -3 \\ 0 & 1 & 0 & -3 & -4 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \\ 7 \end{bmatrix}$$

解得列向量:

$$\begin{bmatrix} 4 \\ 5 \\ 7 \\ 0 \\ 0 \end{bmatrix}$$

- 令方程组后面的  $n - m$  个变量为 0,  $x_j = 0 \quad (j = m + 1, m + 2, \dots, n)$ , 则可得一个初始基可行解:  $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_m^{(0)}, 0, \dots, 0)^T = (b_1, b_2, \dots, b_m, 0, \dots, 0)^T$ .

令非基变量为 0:

$$\begin{bmatrix} 4 \\ 5 \\ 7 \\ 0 \\ 0 \end{bmatrix}$$

### 2.4.2 寻找另一个基可行解：基变换法

为了确定在其他顶点上的可行解，我们需要使用基变换法的方法。

#### 基变换法

当一个基可行解不是最优解或不能判断时，需要过渡到另一个基可行解，即从基可行解，

$$\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_m^{(0)}, 0, \dots, 0)^T$$

对应的可行基

$$B = (p_1, p_2, \dots, p_m)$$

中替换一个列向量，用来替换的列向量与原向量组未被替换的向量线性无关。

例如，用非基变量  $p_{m+t}$  ( $1 \leq t \leq n-m$ ) (称为换入变量) 替换基变量  $p_1$  ( $1 \leq 1 \leq m$ ) (称为换出变量)，就可得到一个新的可行基

$$B_1 = (p_1, \dots, p_{1-1}, p_{m+t}, p_{1+1}, \dots, p_m)$$

从而可以求出一个新的基可行解

$$\mathbf{x}^{(1)} = (x_1^{(1)}, x_2^{(1)}, \dots, x_m^{(1)}, 0, \dots, 0)^T$$

我们仍然用刚才的例子：

$$A' = \begin{bmatrix} 1 & 0 & 0 & -2 & -3 \\ 0 & 1 & 0 & -3 & -4 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

这是初始变换后的矩阵  $A'$ ，前三列已经是单位矩阵。

$$A'' = \begin{bmatrix} 1 & 0 & \textcolor{red}{0} & \textcolor{blue}{-2} & -3 \\ 0 & 1 & \textcolor{red}{0} & \textcolor{blue}{-3} & -4 \\ 0 & 0 & \textcolor{red}{1} & \textcolor{blue}{0} & 0 \end{bmatrix}$$

在这个步骤中，我们交换了  $A'$  的第三列和第四列，粉色为换出变量，蓝色为换入变量。

$$A''' = \begin{bmatrix} 1 & 0 & 0 & * & * \\ 0 & 1 & 0 & * & * \\ 0 & 0 & 1 & * & * \end{bmatrix}$$

通过适当的初等变换（例如对第三列和第四列进行适当的线性组合），我们将前三列转换为单位矩阵，最终得到了这个矩阵。这样我们得到了一个新的可行基，以此可以继续解方程得到另一个可行解。

当然，也有可能换入非基变量后，我们应该首先检查更换后的“基”是否可逆，比如发现新的三个列向量线性相关而不是线性无关，不能构成一个基，那么此时应该换一个非基变量换入。

### Theorem 2.4.1 ▶ 线性规划模型的另一个基可行解

事实上，这个新的基可行解可以用以下公式直接计算出来：

$$\mathbf{x}_i^{(1)} = \begin{cases} \mathbf{x}_i^{(0)} - \theta \beta_{i,m+t}, & i \neq l \\ \theta, & i = l \end{cases} \quad \begin{pmatrix} i = 1, 2, \dots, m, \\ 1 \leq l \leq m, 1 \leq t \leq n-m \end{pmatrix}$$

其中

$$\theta = \frac{\mathbf{x}_i^{(0)}}{\beta_{i,m+t}} = \min_{1 \leq i \leq m} \left\{ \frac{\mathbf{x}_i^{(0)}}{\beta_{i,m+t}} \mid \beta_{i,m+t} > 0 \right\},$$

并且

$$\mathbf{p}_{m+t} = \sum_{i=1}^m \beta_{i,m+t} \mathbf{p}_i.$$

如果  $\mathbf{x}^{(1)} = (\mathbf{x}_1^{(1)}, \mathbf{x}_2^{(1)}, \dots, \mathbf{x}_m^{(1)}, 0, \dots, 0)^T$  仍不是最优解，则可以重复利用这种方法，直到得到最优解为止。

### 2.4.3 最优性检验方法

实际上，我们并不需要把所有的“顶点”都找到再去比较哪个最大。我们可以找到可行域中的任意一点，因为这个点是不独立的（可以由其他所有顶点线性表示），所以如果我们找到了一个顶点  $x_1$ ，得到目标函数值  $z_1 = c x_1$ ；而有任一点  $x = a_1 x_1 + a_2 x_2 + \dots + a_n x_n$ ，得到目标函数值  $z = c x$ 。只需要比较  $z_1$  和  $z$ ，即可知道  $z_1$  是否是最大值，证明过程如下（感兴趣的的同学可以了解）：

将基可行解  $\mathbf{x}^{(1)}$  和这个任意的  $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$  分别代入目标函数得：

$$\begin{aligned} z^{(1)} &= \sum_{i=1}^m c_i x_i^{(1)} = \sum_{i=1}^m c_i b'_i, \psi \\ z &= \sum_{i=1}^n c_i x_i = \sum_{i=1}^m c_i x_i + \sum_{i=m+1}^n c_i x_i \\ &= \sum_{i=1}^m c_i \left( b'_i - \sum_{j=m+1}^n a'_{ij} x_j \right) + \sum_{j=m+1}^n j j^+ j \\ &= \sum_{i=1}^m c_i b'_i + \sum_{j=m+1}^n \left( c_j - \sum_{i=1}^m c_i a'_{ij} \right) x'_j \\ &= z^{(1)} + \sum_{j=m+1}^n (c_j - z_j) x_j \end{aligned}$$

其中  $z_j = \sum_{i=1}^m c_i a'_{ij}$  ( $j = m+1, \dots, n$ )。记  $\sigma_j = c_j - z_j$  ( $j = m+1, \dots, n$ )，则

$$z = z^{(1)} + \sum_{j=m+1}^n O_j x_j$$

注意到：当  $\sigma_j > 0$  ( $j = m+1, \dots, n$ ) 时，就有

$$z > z^{(1)};$$

当  $\sigma_j \leq 0$  ( $j = m+1, \dots, n$ ) 时，就有

$$z \leq z^{(1)}.$$

为此， $\sigma_j = c_j - z_j$  的符号是判别  $\mathbf{x}^{(1)}$  是否为最优解的关键所在，故称之为**检验数**。于是可以得出下面的结论：

### 判断最优解的办法

1. 如果  $\sigma_j \leq 0$  ( $j = m + 1, \dots, n$ ), 则  $x^{(1)}$  是问题的最优解, 最优值为  $z^{(1)}$ ;
2. 如果  $\sigma_j \leq 0$  ( $j = m + 1, \dots, n$ ), 且至少存在一个  $\sigma_{m+k} = 0$  ( $0 \leq k \leq n - m$ ), 则问题有无穷多个最优解,  $x^{(1)}$  是其中之一, 最优值为  $z^{(1)}$ ;
3. 如果  $\sigma_j < 0$  ( $j = m + 1, \dots, n$ ), 则  $x^{(1)}$  是问题的唯一最优解, 最优值为  $z^{(1)}$ ;
4. 如果存在某个检验数  $\sigma_{m+k} > 0$  ( $0 \leq k \leq n - m$ ), 并且对应的系数向量  $p_{m+k}$  的各分量  $a_{i,m+k} \leq 0$  ( $i = 1, 2, \dots, m$ ), 则问题具有无界解 (即无最优解)。

2.4节所有上述所有过程一般上不需要我们了解的过于深入, 因为有现成的计算机函数可以用, 但是可以领会其思想。

### Example 2.4.2 ▶ 单纯形法求解完整过程

问题描述:

$$\min z = 2x_1 + 3x_2$$

$$\text{s.t. } \begin{cases} x_1 + x_2 \geq 7 \\ x_1 - x_2 \leq 10 \\ x_1 \geq 0, \quad x_2 \text{ 自由} \end{cases}$$

步骤 1: 转换为标准形式

1. 处理自由变量: 令  $x_2 = x_2^+ - x_2^-$ , 其中  $x_2^+ \geq 0, x_2^- \geq 0$ 。
2. 引入松弛变量:
  - 约束  $x_1 + x_2 \geq 7 \rightarrow x_1 + x_2^+ - x_2^- - s_1 = 7$  (剩余变量  $s_1 \geq 0$ )。
  - 约束  $x_1 - x_2 \leq 10 \rightarrow x_1 - x_2^+ + x_2^- + s_2 = 10$  (松弛变量  $s_2 \geq 0$ )。
3. 目标函数:

$$z = 2x_1 + 3x_2^+ - 3x_2^- + 0s_1 + 0s_2$$

步骤 2: 初始单纯形表

选择初始基变量  $\{x_1, s_2\}$ , 通过行变换使基变量对应的列变为单位矩阵:

	$x_1$	$x_2^+$	$x_2^-$	$s_1$	$s_2$	RHS
$x_1$	1	1	-1	-1	0	7
$s_2$	1	-1	1	0	1	10

行变换: Row 2  $\leftarrow$  Row 2 - Row 1

更新后的单纯形表:

	$x_1$	$x_2^+$	$x_2^-$	$s_1$	$s_2$	RHS
$x_1$	1	1	-1	-1	0	7
$s_2$	0	-2	2	1	1	3

**步骤 3: 计算检验数**

$$\text{检验数公式: } \sigma_j = c_j - c_B^T A_j$$

当前基变量  $\{x_1, s_2\}$ ,  $c_B = [2, 0]$ 。

$$\begin{aligned}\sigma_{x_1} &= 2 - [2, 0] \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0 \\ \sigma_{x_2^+} &= 3 - [2, 0] \begin{bmatrix} 1 \\ -2 \end{bmatrix} = 1 \\ \sigma_{x_2^-} &= -3 - [2, 0] \begin{bmatrix} -1 \\ 2 \end{bmatrix} = -1 \quad (\text{最小, 进基}) \\ \sigma_{s_1} &= 0 - [2, 0] \begin{bmatrix} -1 \\ 1 \end{bmatrix} = 2 \\ \sigma_{s_2} &= 0 - [2, 0] \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0\end{aligned}$$

**步骤 4: 确定进基和出基变量**

- 进基变量:  $x_2^-$  (因  $\sigma_{x_2^-} = -1 < 0$ )。
- 出基变量: 计算比率  $\theta = \frac{\text{RHS}}{\text{进基列, 此处为}[-1, 2]}$ :
  - $\theta_{x_1} = \frac{7}{-1}$  (舍去, 因分母为负)。
  - $\theta_{s_2} = \frac{3}{2} = 1.5$  (最小正值, 出基)。

**步骤 5: 更新单纯形表**

通过行变换使  $x_2^-$  对应的列变为  $[0, 1]^T$ :

1. Row 2  $\leftarrow$  Row 2 / 2
2. Row 1  $\leftarrow$  Row 1 + Row 2

更新后的单纯形表:

	$x_1$	$x_2^+$	$x_2^-$	$s_1$	$s_2$	RHS
$x_1$	1	0	0	-0.5	0.5	8.5
$x_2^-$	0	-1	1	0.5	0.5	1.5

步骤 6: 重新计算检验数

当前基变量  $\{x_1, x_2^-\}$ ,  $c_B = [2, -3]$ 。

$$\begin{aligned}\sigma_{x_1} &= 2 - [2, -3] \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0 \\ \sigma_{x_2^+} &= 3 - [2, -3] \begin{bmatrix} 0 \\ -1 \end{bmatrix} = 0 \\ \sigma_{x_2^-} &= -3 - [2, -3] \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0 \\ \sigma_{s_1} &= 0 - [2, -3] \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix} = 2.5 \\ \sigma_{s_2} &= 0 - [2, -3] \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = 0.5\end{aligned}$$

所有检验数  $\geq 0$ , 当前解为最优解。

步骤 7: 提取最优解

从最终单纯形表:

- $x_1 = 8.5, x_2^- = 1.5, x_2^+ = s_1 = s_2 = 0$ 。
- 原变量  $x_2 = x_2^+ - x_2^- = -1.5$ 。
- 目标值  $z = 2 \times 8.5 + 3 \times (-1.5) = 12.5$ 。

最终答案:

- 最优解:  $x_1 = 8.5, x_2 = -1.5$ 。
- 最优值:  $z = 12.5$ 。
- 验证: 与图解法结果一致, 求解正确。

## 2.5 线性规划问题的灵敏度分析

在线性规划模型

$$\max z = \mathbf{c} \cdot \mathbf{x}$$

约束条件为

$$\begin{cases} \mathbf{Ax} = \mathbf{b} \\ \mathbf{x} \geq \mathbf{0} \end{cases}$$

其中，总假设  $\mathbf{A}, \mathbf{b}, \mathbf{c}$  都是常数，但这些数值在许多情况下是由试验或测量得到的，特别是在迭代计算中，这些数值都是近似值。通常， $\mathbf{A}$  表示工艺条件， $\mathbf{b}$  表示资源条件， $\mathbf{c}$  表示市场条件。在实际中，可能有多种原因引起它们的变化。

现在的问题是：这些系数在什么范围内变化时，线性规划问题的最优解不发生变化？这就是灵敏度分析要研究的问题。

这一问题比较复杂，本课程不做深入探究，但是大作业可以考虑

## 2.6 应用案例分析

### 2.6.1 下料问题

#### Example 2.6.1 ▶ 下料问题

某单位需要加工作 100 套工架，每套工架需用长为 2.9m, 2.1m 和 1.5m 的圆钢各一根，已知原材料长 7.4m，现在的问题是如何下料使得所用的原材料最省？

解：简单分析，在每一根原材料上各截取一根 2.9m, 2.1m 和 1.5m 的圆钢做成一套工架，每根原材料剩下料头 0.9m。要完成 100 套工架，就需要用 100 根原材料，共剩余 90m 料头。

若采用套裁方案，则可以节省原材料。下面给出了几种可能的套裁方案，如表 2.1 所示。

实际上，为了保证完成这 100 套工架，使所用原材料最省，可以混合使用各种下料方案。

设按方案 A, B, C, D, E 下料的原材料数分别为  $x_1, x_2, x_3, x_4, x_5$ 。根据表格 2-1，可以得到如下线性规划模型。目标函数：

$$\min z = 0x_1 + 0.1x_2 + 0.2x_3 + 0.3x_4 + 0.8x_5$$

约束条件：

$$\begin{cases} x_1 + 2x_2 + x_4 = 100 \\ 2x_3 + 2x_4 + x_5 = 100 \\ 3x_1 + x_2 + 2x_3 + 3x_5 = 100 \\ x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

所有 2.9 的料总数 100，所有 2.1 的料总数 100，所有 1.5 的料总数 100。

长度/m	方案				
	A	B	C	D	E
2.9	1	2	0	1	0
2.1	0	0	2	2	1
1.5	3	1	2	0	3
合计/m	7.4	7.3	7.2	7.1	6.6
料头/m	0	0.1	0.2	0.3	0.8

Table 2.1: 例 2.6.1 下料方案

### Code Snippet 2.6.2 ▶ MATLAB 代码

```

1 c = [0, 0.1, 0.2, 0.3, 0.8]';
2 b1 = [0, 0, 0, 0, 0]';
3 b2 = [100, 100, 100]';
4 A1 = [-1, 0, 0, 0, 0; 0, -1, 0, 0, 0; 0, 0, -1, 0, 0; 0, 0, 0, -1, 0; 0,
       0, 0, 0, -1]';
5 A2 = [1, 2, 0, 1, 0; 0, 0, 2, 2, 1; 3, 1, 2, 0, 3]';
6 [x, fv] = linprog(c, A1, b1, A2, b2);

```

运行该程序后，立即可以得到最优解为  $\mathbf{x} = (12.8243, 27.1757, 17.1757, 32.8243, 0)^T$ 。四舍五入的方法取整得  $\mathbf{x} = (13, 27, 17, 33, 0)^T$ 。最优值为  $z = 16$ ，即接方案 A 下料 13 根，方案 B 下料 27 根，方案 C 下料 17 根，方案 D 下料 33 根，共需原材料 90 根就可以制作完成 100 套工架，剩余料头最少为 16m。

### matlab 代码解析

#### 1. 目标函数定义:

$c = [0, 0.1, 0.2, 0.3, 0.8]'$  定义了最小化料头长度的目标函数系数向量，对应五种下料方案的料头长度。

#### 2. 约束条件设置:

- $b1 = [0, 0, 0, 0, 0]'$  设置变量非负约束的右端项
- $b2 = [100, 100, 100]'$  设置三种圆钢需求量的右端项
- $A1$  矩阵通过负单位矩阵实现  $x_i \geq 0$  的非负约束
- $A2$  矩阵的每一列对应一个下料方案:
  - 第一行: 2.9m 圆钢的生产数量约束
  - 第二行: 2.1m 圆钢的生产数量约束
  - 第三行: 1.5m 圆钢的生产数量约束

#### 3. 线性规划求解:

`linprog(c, A1, b1, A2, b2)` 调用 MATLAB 线性规划求解器，其中：

- 输入参数：目标系数  $c$ ，不等式约束  $A1, b1$ ，等式约束  $A2, b2$
- 输出参数： $x$  为最优解向量， $fv$  为最优目标值

#### 4. 结果修正:

原始解包含小数，通过四舍五入得到整数解  $(13, 27, 17, 33, 0)$ ，此时总用料 90 根，剩余料头 16m。

### Code Snippet 2.6.3 ▶ LINGO 代码

```

1 MODEL
2 sets;
3 row/1 2 3/:b;
4 arrange/1..5/:x c;
5 endsets;
6
7 data:
8 b=100,100,100;
9 c=0 0 1 0.2 0 3 0.8.
10 a=1,2,0,1,0,0,0,2,2,1 3,1 2,0,3;
11 enddata.
12
13 [oBJ] min=@sum(arrange(j):c(j)*x(j));

```

```

14 @for(row(i):@sum(arrange(j);a(lj)*x(j))=b(i););
15 @for(arrange(j);x(j)>=0););
16 END

```

运行该程序后，立即可以得到最优解为  $x = (0, 40, 30, 20, 0)^T$ ，最优值为  $z = 16$ ，即按方案 B 下料 40 根，方案 C 下料 30 根，方案 D 下料 20 根，共需原材料 90 根就可以制作完成 100 套工架，剩余料头最少为 16m。

## 2.6.2 连续投资问题

### Example 2.6.4 ▶ 连续投资问题

某投资公司拟制定今后 5 年的投资计划，初步考虑下面的四个投资项目：

项目 A：从第 1 年到第 4 年每年年初需要投资，于次年年末收回成本，并可获利润 15%；

项目 B：第 3 年年初需要投资，到第 5 年年末可以收回成本，并获得利润 25%，但为了保证足够的资金流动，规定该项目的投资金额上限为不超过总金额的 40%

项目 C：第 2 年年初需要投资，到第 5 年年末可以收回成本，并获得利润 40%，但规定该项目的最大投资金额不超过总金额的 30%；

项目 D：5 年内每年年初可以购买公债，于当年年末可以归还本金，并获利息 6%。

该公司现有投资金额 100 万元，请你帮助该公司制定这些项目每年的投资计划，使公司到第 5 年年末能够获得最大的利润。

解：虽然这是一个连续投资问题，即属于动态优化问题，但是在这里可以用静态优化的方法来解决。用决策变量分别表示第  $i$  年年初为项目 A, B, C, D 的投资额，根据问题的要求各变量的对应关系如表 2-7 所示（表格待补充），表中空白处表示当年不能为该项目投资，也可认为投资额为 0。

首先注意到，项目 D 每年都可以投资，并且当年末就能收回本息，所以公司每年应把全部资金都投出去。因此，投资方案应满足下面的条件。第 1 年：将 100 万元资金全部用于项目 A 和项目 D 的投资，即：

$$x_{11} + x_{14} = 1000000$$

第 2 年：因为第 1 年用于项目 A 的投资到第 2 年年末才能收回，所以能用于第 2 年年初的投资金额只有项目 D 的第 1 年收回的本息总额  $x_{14}(1 + 0.06)$ 。于是第 2 年的投资

分配为：

$$x_{21} + x_{23} + x_{24} = 1.06x_{14}$$

于是可以得到问题的线性规划模型为：

$$\max z = 1.15x_{41} + 1.25x_{32} + 1.40x_{23} + 1.06x_{54}$$

约束条件如下：

$$\begin{aligned} x_{11} + x_{14} &= 1000000 \\ -1.06x_{14} + x_{21} + x_{23} + x_{24} &= 0 \\ -1.15x_{11} - 1.06x_{24} + x_{31} + x_{32} + x_{34} &= 0 \\ -1.15x_{21} - 1.06x_{34} + x_{41} + x_{44} &= 0 \\ -1.15x_{31} - 1.06x_{44} + x_{54} &= 0 \\ x_{32} &\leq 400000 \\ x_{32} &\leq 300000 \\ x_{33} \leq x_{33} + x_{44} &\leq 0.3 \\ 4 \leq t &\leq 5 \end{aligned}$$

考虑到这个问题的实际情况，这里使用 LINGO 求解该线性规划模型。

运行该程序后，得到最优解：

$$x_{11} = 716981.1, x_{14} = 283018.9, x_{23} = 300000, x_{31} = 424528.3, x_{32} = 400000, x_{54} = 488207.5$$

其他的变量均为零，最优值为：

$$z = 1437500$$

即连续投资方案为：

- 第 1 年用于投资项目 A 的金额为 716981.1 元，项目 D 的金额为 283018.9 元。
- 第 2 年用于项目 C 的投资金额为 300000 元。
- 第 3 年用于项目 A 的投资为 424528.3 元，项目 B 的金额为 400000 元。
- 第 5 年用于投资项目 D 的金额为 488207.5 元。

到第 5 年年末，该公司拥有总资金为 1437500 元，收益率为 43.75%。

## 2.7 第二章作业

### 2.7.1 配餐问题

**Q:** 编程求解本章 ppt 中例 2.2 合理配餐问题，需要的基础数据自拟。

某幼儿园为了保证孩子们的健康成长，要求对每天的膳食进行合理科学的搭配，以保证孩子们对各种营养的需求。从营养学的角度。假设共有 5 种食品  $A_j(j = 1, 2, \dots, 6)$  可供选择，每种食品都含有加 6 种不同的营养成分  $B_i(i = 1, 2, \dots, 6)$ 。而且每单位的食品  $A_j$  含有营养成分  $B_i$  的含量如下表所示(数据为自拟)：

营养成分	食品					最低需求量
	A1	A2	A3	A4	A5	
B1	4.0	0.4	0.8	0.5	0.9	16.0
B2	0.5	4.0	0.5	0.7	0.7	26.0
B3	0.6	0.2	4.0	0.4	0.5	18.0
B4	0.7	0.1	0.3	4.0	0.3	12.0
B5	0.8	0.9	0.2	0.3	4.0	14.0
B6	1.2	1.3	1.4	1.4	1.3	20.0
食品单价	5	6	7	8	9	
摄入量最小值	2.0	3.0	3.0	1.0	3.0	

Table 2.2: 营养数据表

1. 每人每天对营养成分  $B_i$  的最低需求为  $b_i(i = 1, 2, \dots, 6)$ ，而且食品  $A_j$  的单价为  $c_j(j = 1, 2, \dots, 5)$ 。问如何合理科学地制定配餐方案，既可以保证孩子们的营养需求，又使每人每天所需的费用最低？
2. 除了如上的要求之外，如果还要求各种食品的合理搭配，即要求每人每天对食品  $A_j$  的摄入量不少于  $d_j(j = 1, 2, \dots, 5)$ ，问配餐方案又如何？

**A:** 分析如下。

1. 基础配餐问题（仅考虑营养需求）

- **决策变量：**设每天采购食品  $A_j$  的数量为  $x_j$  (单位：份)，其中  $j = 1, 2, \dots, 5$

- 目标函数: 最小化总费用

$$\min z = 5x_1 + 6x_2 + 7x_3 + 8x_4 + 9x_5$$

- 约束条件:

(a) 营养成分需求 (满足最低摄入量):

$$\begin{cases} 4.0x_1 + 0.4x_2 + 0.8x_3 + 0.5x_4 + 0.9x_5 \geq 16.0 & (\text{营养成分 } B_1) \\ 0.5x_1 + 4.0x_2 + 0.5x_3 + 0.7x_4 + 0.7x_5 \geq 26.0 & (\text{营养成分 } B_2) \\ 0.6x_1 + 0.2x_2 + 4.0x_3 + 0.4x_4 + 0.5x_5 \geq 18.0 & (\text{营养成分 } B_3) \\ 0.7x_1 + 0.1x_2 + 0.3x_3 + 4.0x_4 + 0.3x_5 \geq 12.0 & (\text{营养成分 } B_4) \\ 0.8x_1 + 0.9x_2 + 0.2x_3 + 0.3x_4 + 4.0x_5 \geq 14.0 & (\text{营养成分 } B_5) \\ 1.2x_1 + 1.3x_2 + 1.4x_3 + 1.4x_4 + 1.3x_5 \geq 20.0 & (\text{营养成分 } B_6) \end{cases}$$

(b) 非负约束:

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

## 2. 扩展配餐问题 (增加食品摄入量约束)

- 决策变量: 同上, 仍为  $x_j$
- 目标函数: 同上, 仍为最小化总费用

$$\min z = 5x_1 + 6x_2 + 7x_3 + 8x_4 + 9x_5$$

- 约束条件:

(a) 原营养成分需求: 同上

(b) 食品摄入量下限 (表格中“摄入量最值”):

$$\begin{cases} x_1 \geq 2.0 \\ x_2 \geq 3.0 \\ x_3 \geq 3.0 \\ x_4 \geq 1.0 \\ x_5 \geq 3.0 \end{cases}$$

(c) 非负约束: 同上

### Code Snippet 2.7.1 ▶ Matlab 代码

```
1 % 配餐优化: 求解浮点数和整数解
2 clear; clc;
3
4 % 营养含量矩阵 A (6x5, 行: B1-B6, 列: A1-A5)
5 A = [4.0, 0.4, 0.8, 0.5, 0.9;
6     0.5, 4.0, 0.5, 0.7, 0.7;
7     0.6, 0.2, 4.0, 0.4, 0.5;
8     0.7, 0.1, 0.3, 4.0, 0.3;
9     0.8, 0.9, 0.2, 0.3, 4.0;
10    1.2, 1.3, 1.4, 1.4, 1.3];
11
12 % 最低营养需求 B (6x1)
13 B = [16.0; 26.0; 18.0; 12.0; 14.0; 20.0];
14
15 % 食物单价 c (5x1)
16 c = [5; 6; 7; 8; 9];
17
18 % 约束: A*x >= B => -A*x <= -B
19 A_ineq = -A;
20 b_ineq = -B;
21
22 % 第一部分: 仅满足营养需求
23
24 % 下界
25 lb1 = zeros(5,1);
26
27 % 求解浮点数解
28 [x1, fval1, exitflag1] = linprog(c, A_ineq, b_ineq, [], [], lb1);
29
30 % 输出浮点数结果
31 fprintf('第一部分 (浮点数解):\n');
32 if exitflag1 > 0
```

```
33     fprintf('摄入量: A1=%.2f, A2=%.2f, A3=%.2f, A4=%.2f, A5=%.2f\n',
34         ↵ x1);
35     fprintf('成本: %.2f\n', fval1);
36 else
37     fprintf('未找到解\n');
38 end
39 % 求解整数解
40 intcon = 1:5;
41 [x1_int, fval1_int, exitflag1_int] = intlinprog(c, intcon, A_ineq,
42         ↵ b_ineq, [], [], lb1, []);
43 % 输出整数结果
44 fprintf('\n 第一部分 (整数解) :\n');
45 if exitflag1_int > 0
46     fprintf('摄入量: A1=%d, A2=%d, A3=%d, A4=%d, A5=%d\n', x1_int);
47     fprintf('成本: %.2f\n', fval1_int);
48     nutrition1_int = A * x1_int;
49     fprintf('营养验证:\n');
50     for i = 1:6
51         fprintf('B%d: %.2f >= %.2f (%s)\n', i, nutrition1_int(i), B(i),
52             ↵ ...
53             '满足', '不满足');
54     end
55 else
56     fprintf('未找到解\n');
57 end
58 % 第二部分: 增加最低摄入量约束
59
60 % 最低摄入量 d (5x1)
61 d = [2.0; 3.0; 3.0; 1.0; 3.0];
62
63 % 下界
64 lb2 = d;
```

```
65
66 % 求解浮点数解
67 [x2, fval2, exitflag2] = linprog(c, A_ineq, b_ineq, [], [], lb2);
68
69 % 输出浮点数结果
70 fprintf('\n 第二部分（浮点数解）:\n');
71 if exitflag2 > 0
72     fprintf('摄入量: A1=% .2f, A2=% .2f, A3=% .2f, A4=% .2f, A5=% .2f\n',
73             ↵ x2);
74     fprintf('成本: %.2f\n', fval2);
75 else
76     fprintf('未找到解\n');
77 end
78
79 % 求解整数解
80 lb2_int = ceil(d);
81 A_ineq_int = [A_ineq; zeros(1,5)];
82 A_ineq_int(end,4) = -1;
83 b_ineq_int = [b_ineq; -2];
84 [x2_int, fval2_int, exitflag2_int] = intlinprog(c, intcon, A_ineq_int,
85         ↵ b_ineq_int, [], [], lb2_int, []);
86
87 % 输出整数结果
88 fprintf('\n 第二部分（整数解）:\n');
89 if exitflag2_int > 0
90     fprintf('摄入量: A1=%d, A2=%d, A3=%d, A4=%d, A5=%d\n', x2_int);
91     fprintf('成本: %.2f\n', fval2_int);
92     nutrition2_int = A * x2_int;
93     fprintf('营养验证:\n');
94     for i = 1:6
95         fprintf('B%d: %.2f >= %.2f (%s)\n', i, nutrition2_int(i), B(i),
96             ↵ ...
97             '满足', '不满足');
98     end
99     fprintf('最低摄入量验证:\n');
```

```

97     for j = 1:5
98         fprintf('A%d: %d >= %.1f (%s)\n', j, x2_int(j), d(j), ...
99             '满足', '不满足');
100    end
101 else
102     fprintf('未找到解\n');
103 end

```

实验通过 MATLAB 代码求解配餐优化问题，得到第一部分（仅满足营养需求）和第二部分（满足营养需求及最低摄入量）的浮点数解与整数解。所有解均经过验证，满足营养需求  $B_i$  (16.0, 26.0, 18.0, 12.0, 14.0, 20.0) 及第二部分的最低摄入量  $d_j$  (2.0, 3.0, 3.0, 1.0, 3.0)。结果如下：

Table 2.3: 配餐优化结果

部分	解类型	摄入量 ( $A_1, A_2, A_3, A_4, A_5$ )	成本
第一部分	浮点数解	3.64, 5.06, 3.35, 1.89, 1.32	99.02
	整数解	3, 5, 4, 2, 2	107.00
第二部分	浮点数解	2.00, 4.94, 3.37, 2.05, 3.00	106.67
	整数解	2, 5, 4, 2, 3	111.00

程序采用 MATLAB 实现配餐优化，思路如下：

- **数据定义：** 定义营养含量矩阵  $A$ 、最低需求  $B$ 、单价  $c$ 、最低摄入量  $d$ ，构建约束  $A \cdot x \geq B$ 。
- **浮点数解：** 使用 `linprog` 求解线性规划问题，最小化成本  $c^T \cdot x$ ，满足营养需求和非负约束（第一部分）或最低摄入量约束（第二部分）。
- **整数解：** 使用 `intlinprog` 求解整数线性规划，添加整数约束  $x_j \in \mathbb{Z}$ ，并在第二部分确保  $x_j \geq [d_j]$ 。额外约束  $x_4 \geq 2$  保证 B4 满足。
- **验证：** 计算  $A \cdot x$  和  $x_j \geq d_j$ ，验证所有约束满足情况，确保解的可行性。

### 2.7.2 战略轰炸问题

**Q:** 某战略轰炸机群奉命摧毁敌人军事目标，已知该目标有四个要害部位，只要摧毁其中之一即可达到目的。为完成此项轰炸任务的汽油消耗量限制为 48000L，重型炸弹 48 枚，轻型炸弹 32 枚。飞机携带重型炸弹时每升汽油可飞行 2km，带轻型炸弹时每升汽油可飞行 3km，空载时每升汽油可飞行 4km。又知每架飞机每次只能装载一枚炸弹，每起飞轰炸一次除来回路途汽油消耗外，起飞和降落每次消耗 100L 汽油，其他相关数据如表所示。为了保证以最大的可能性摧毁敌方军事目标，应该如何确定飞机的轰炸方案。

敌要害部位	距离机场距离 (km)	每枚重型炸弹摧毁概率	每枚轻型炸弹摧毁概率
1	450	0.10	0.08
2	480	0.20	0.16
3	540	0.15	0.12
4	600	0.25	0.20

Table 2.4: 轰炸目标数据表

**A:** 分析如下。

#### 问题分析

- 目标：最大化摧毁敌方军事目标（四个要害部位中至少一个）的可能性。
- 资源限制：
  - 总燃油：48000L。
  - 重型炸弹：48 枚。
  - 轻型炸弹：32 枚。
  - 每架飞机每次任务仅携带一枚炸弹（重型或轻型）。
  - 每任务（包括起飞和降落）额外消耗 100L 燃油。
- 燃油效率：
  - 重型炸弹：2 km/L。
  - 轻型炸弹：3 km/L。

- 空载: 4 km/L (本问题不涉及空载)。
- 数据:
  - 四个要害部位, 距离机场分别为 450km、480km、540km、600km。
  - 每部位被重型或轻型炸弹摧毁的概率如表所示。

## 数学模型

### 决策变量:

- $x_{i,h}$ : 对第  $i$  个部位使用重型炸弹的任务数 (整数,  $i = 1, 2, 3, 4$ )。
- $x_{i,l}$ : 对第  $i$  个部位使用轻型炸弹的任务数 (整数,  $i = 1, 2, 3, 4$ )。

### 目标函数:

- 最大化总期望成功数:

$$\max z = 0.10x_{1,h} + 0.08x_{1,l} + 0.20x_{2,h} + 0.16x_{2,l} + 0.15x_{3,h} + 0.12x_{3,l} + 0.25x_{4,h} + 0.20x_{4,l}$$

### 约束条件:

#### 1. 燃油约束:

- 重型炸弹任务到部位  $i$  的燃油消耗: 来回距离  $2 \cdot d_i$  km, 效率 2 km/L, 燃油  $d_i$  L, 加 100L 起降, 总计  $d_i + 100$  L。
- 轻型炸弹任务: 来回距离  $2 \cdot d_i$  km, 效率 3 km/L, 燃油  $\frac{2 \cdot d_i}{3}$  L, 加 100L 起降, 总计  $\frac{2 \cdot d_i}{3} + 100$  L。
- 总燃油:

$$550x_{1,h} + 400x_{1,l} + 580x_{2,h} + 420x_{2,l} + 640x_{3,h} + 460x_{3,l} + 700x_{4,h} + 500x_{4,l} \leq 48000$$

#### 2. 重型炸弹约束:

$$x_{1,h} + x_{2,h} + x_{3,h} + x_{4,h} \leq 48$$

#### 3. 轻型炸弹约束:

$$x_{1,l} + x_{2,l} + x_{3,l} + x_{4,l} \leq 32$$

#### 4. 非负性和整数约束:

$$x_{i,h}, x_{i,l} \geq 0, \quad x_{i,h}, x_{i,l} \in \mathbb{Z}, \quad \forall i = 1, 2, 3, 4$$

矩阵形式:

- 定义变量向量:

$$\mathbf{x} = [x_{1,h}, x_{1,l}, x_{2,h}, x_{2,l}, x_{3,h}, x_{3,l}, x_{4,h}, x_{4,l}]^T$$

- 目标函数:

$$\max z = \mathbf{c}^T \mathbf{x}, \quad \mathbf{c} = [0.10, 0.08, 0.20, 0.16, 0.15, 0.12, 0.25, 0.20]^T$$

- 燃油约束:

$$\mathbf{a}_{\text{fuel}}^T \mathbf{x} \leq 48000, \quad \mathbf{a}_{\text{fuel}} = [550, 400, 580, 420, 640, 460, 700, 500]^T$$

- 炸弹约束:

$$\mathbf{A}_{\text{bomb}} \mathbf{x} \leq \mathbf{b}_{\text{bomb}}, \quad \mathbf{A}_{\text{bomb}} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{b}_{\text{bomb}} = [48, 32]^T$$

- 非负性和整数约束:

$$\mathbf{x} \geq \mathbf{0}, \quad \mathbf{x} \in \mathbb{Z}^8$$

# 运输规划

如果有若干个产地，同时又有若干个销售地。那么，根据已有的交通网，应如何制定“将产品从产地运到销售地”的调运方案，使总的运输费用最少，或运输路线最短？运筹问题的数学模型就是运输规划模型。事实上，运输规划是一类特殊的线性规划。

## 3.1 运输问题

### 3.1.1 产销平衡

#### Example 3.1.1 ► 产销平衡的运输问题

例：已知有  $m$  个工厂  $A_i (i = 1, 2, \dots, m)$ ，其供应量（产量）分别为  $a_i (i = 1, 2, \dots, m)$ ，有  $n$  个销售地  $B_j (j = 1, 2, \dots, n)$ ，其需要量分别为  $b_j (j = 1, 2, \dots, n)$ 。从  $A_i$  到  $B_j$  运输单位物资的运价（单价）为  $c_{ij} (i = 1, 2, \dots, m; j = 1, 2, \dots, n)$ 。假设总产量等于总销量，且产销是平衡的，其数据如表 3.1 所示。

产品	销量				产量
	$B_1$	$B_2$	$\cdots$	$B_n$	
$A_1$	$c_{11}$	$c_{12}$	$\cdots$	$c_{1n}$	$a_1$
$A_2$	$c_{21}$	$c_{22}$	$\cdots$	$c_{2n}$	$a_2$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$A_m$	$c_{m1}$	$c_{m2}$	$\cdots$	$c_{mn}$	$a_m$
销量	$b_1$	$b_2$	$\cdots$	$b_n$	

Table 3.1: 相关数据表

解：如采用  $x_{ij}$  表示从  $A_i$  到  $B_j (i = 1, 2, \dots, m; j = 1, 2, \dots, n)$  的运量，则么在产销平衡的情况下，要求供给总量等于最小的调运方式。

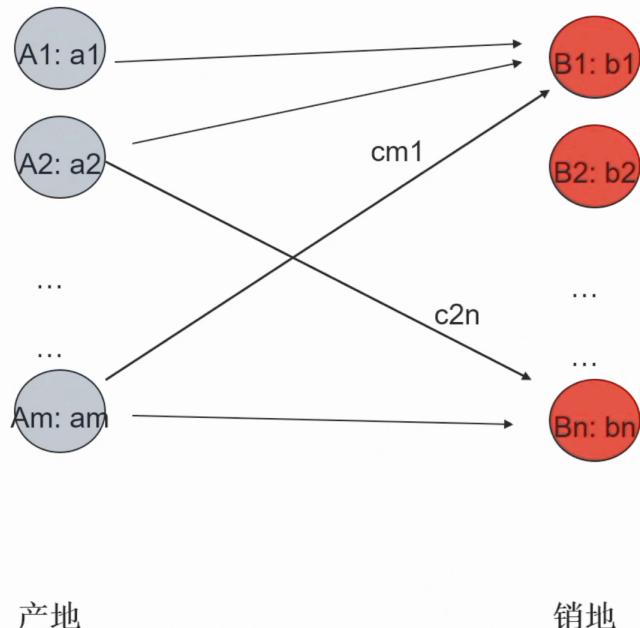


Figure 3.1: 例 1.1.3 图解

优化目标为最小化总运费，数学模型如下：

$$\min z = \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} \quad (3.1)$$

约束条件:

$$\sum_{j=1}^n x_{ij} = a_i, \quad i = 1, 2, \dots, m \quad (\text{m 个约束方程})$$

$$\sum_{i=1}^m x_{ij} = b_j, \quad j = 1, 2, \dots, n \quad (\text{n 个约束方程})$$

$$x_{ij} \geq 0, \quad i = 1, 2, \dots, m; j = 1, 2, \dots, n \quad (\text{m} \times n \text{ 个变量})$$

约束条件的系数矩阵共  $m + n$  行,  $m \times n$  列的矩阵, 即  $x_{i,j}$  的系数向量。

$$\mathbf{p}_{ij} = (0, \dots, 1, \dots, 1, \dots, 0)^\top, \quad i \text{ 行}, \quad m+j \text{ 行} \quad (3.2)$$

分量中除第  $i$  个和第  $m+j$  个元素为 1，其余均为 0。

关于产销平衡的运输问题，还有

$$\sum_{i=1}^m a_i = \sum_{i=1}^m \left( \sum_{j=1}^n x_{ij} \right) = \sum_{j=1}^n \left( \sum_{i=1}^m x_{ij} \right) = \sum_{j=1}^n b_j, \quad (3.3)$$

所以模型中最多有  $m+n-1$  个独立约束方程，即系数矩阵的秩不超过  $m+n-1$ <sup>a</sup>。

<sup>a</sup>原本有  $m+n$  个方程，但是根据题目中提到的“产销平衡”，可以列出方程 3.3，即自由度减 1，消去了一个约束，可行域增大

### 3.1.2 产销不平衡

事实上，我们知道，市场上几乎不可能出现“产销平衡”的情况，供大于求（产大于销）、供不应求（销大于产）的情况更为常见。**产销不平衡问题一般都是转化为产销平衡问题解决的。**

#### Example 3.1.2 ▶ 产销不平衡的运输问题

(1) 产大于销时：

由于

$$\sum_{j=1}^n b_j < \sum_{i=1}^m a_i, \quad (3.4)$$

则问题的模型为

$$\min z = \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij}, \quad (3.5)$$

约束条件：

$$\sum_{j=1}^n x_{ij} \leq a_i \quad i = 1, 2, \dots, m, \quad (3.6)$$

$$\sum_{i=1}^m x_{ij} = b_j \quad j = 1, 2, \dots, n, \quad (3.7)$$

$$x_{ij} \geq 0 \quad i = 1, 2, \dots, m; j = 1, 2, \dots, n. \quad (3.8)$$

其中， $\sum_{j=1}^n x_{ij} \leq a_i$  表示  $i$  产品的总产量小于或等于其产量。

(2) 销大于产时：

由于

$$\sum_{j=1}^n b_j > \sum_{i=1}^m a_i, \quad (3.9)$$

则问题的数学模型为

$$\min z = \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij}, \quad (3.10)$$

约束条件:

$$\sum_{j=1}^n x_{ij} = a_i \quad i = 1, 2, \dots, m, \quad (3.11)$$

$$\sum_{i=1}^m x_{ij} \leq b_j \quad j = 1, 2, \dots, n, \quad (3.12)$$

$$x_{ij} \geq 0 \quad i = 1, 2, \dots, m; j = 1, 2, \dots, n. \quad (3.13)$$

其中,  $\sum_{i=1}^m x_{ij} \leq b_j$  表示  $i$  产品的总产量小于或等于其销量。

此时, 要将各约束资源

$$\sum_{i=1}^m a_i - \sum_{j=1}^n b_j = b_{n+1}, \quad (3.14)$$

在生产地或销地储存起来, 即使没有一个虚拟的销售地, 其运费为零, 即设  $x_{i,n+1}$  表示产地  $A_i$  多年产 (需要储存) 的物资数量, 运费为  $c_{i,n+1} = 0$  ( $i = 1, 2, \dots, m$ ), 其目标函数不变。于是问题的模型变为

$$\min z = \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij}, \quad (3.15)$$

约束条件:

$$\sum_{j=1}^n x_{ij} + x_{i,n+1} = a_i \quad (i = 1, 2, \dots, m), \quad (3.16)$$

$$\sum_{i=1}^m x_{ij} = b_j \quad (j = 1, 2, \dots, n), \quad (3.17)$$

$$\sum_{i=1}^m x_{i,n+1} = \sum_{i=1}^m a_i - \sum_{j=1}^n b_j = b_{n+1}, \quad (3.18)$$

$$x_{ij}, x_{i,n+1} \geq 0 \quad (i = 1, 2, \dots, m; j = 1, 2, \dots, n). \quad (3.19)$$

公式 (3.16) 表示转化为产销平衡问题。

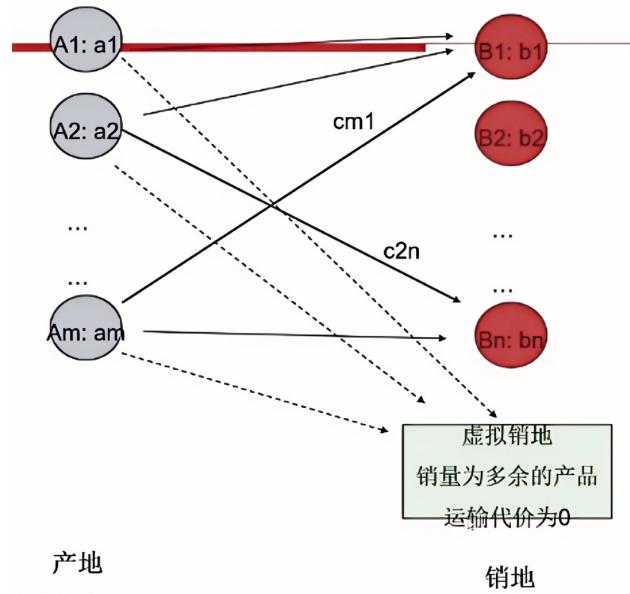


Figure 3.2: 产大于销的情况

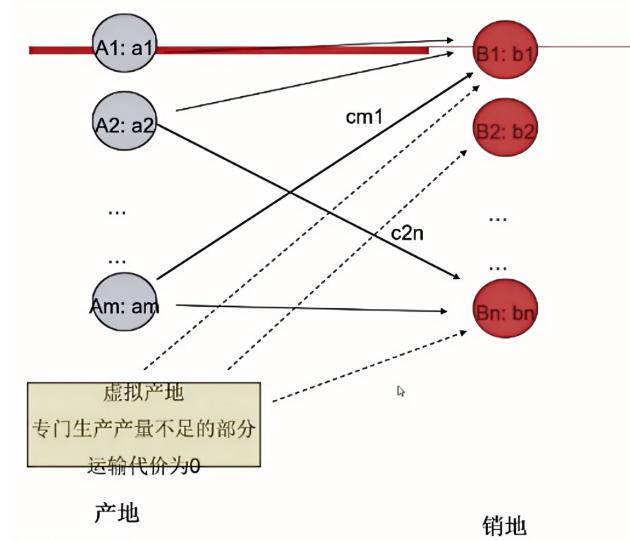


Figure 3.3: 销大于产的情况

# 整数规划

## 4.1 整数规划问题与数学模型

### 4.1.1 整数规划问题的定义

在前面的线性规划和运输规划问题中，最优解一般都是实数。但对于实际中的具体问题的解常常要求必须取整数，即称为整数解，例如，问题答案是几个人、几台设备、几辆车等，无法用实数表达。因此，对于要求最优整数解的问题，就涉及到整数规划。

#### Definition 4.1.1 ▶ 整数规划

如果一个数学规划的某些决策变量或全部决策变量要求必须取整数，则这样的问题称为整数规划问题；相应的模型称为整数规划模型。

- 纯整数规划问题：所有的决策变量都为非负整数的整数规划；
- 混合整数规划问题：存在决策变量为负整数的整数规划；
- 0-1 规划：所有的决策变量只能取0 或 1 的整数规划；

### 4.1.2 整数规划问题的数学模型

#### Theorem 4.1.2 ▶ 一般形式的整数线性规划问题

$$\max(\min) z = \sum_{j=1}^n c_j x_j$$

s.t.

$$\begin{cases} \sum_{j=1}^n a_{ij} x_j \leq (\geq, =) b_i, & (i = 1, 2, \dots, m), \\ x_j \geq 0, & x_j \text{ 为整数} \quad (j = 1, 2, \dots, n). \end{cases}$$

**Example 4.1.3 ▶ 产品生产（纯整数规划问题）**

例：某厂生产  $A_1$  和  $A_2$  两种产品，需要经过  $B_1$ 、 $B_2$ 、 $B_3$  三道工序加工。单件工时和利润值以及各工序每月工时定额表 4.1。问工厂应如何安排生产才能使总利润最大？

	$B_1$	$B_2$	$B_3$	利润（元/件）
$A_1$	0.3	0.2	0.3	25
$A_2$	0.7	0.1	0.5	40
工时定额（小时/月）	250	100	150	

Table 4.1: 工厂加工条件与利润

解：根据表 2-1 的最后一栏的利润数据，生产  $A_1$  件、 $A_2$  件能获取的总利润为  $25x_1 + 40x_2$ ，因此，该问题的数学模型为：

$$\max \quad 25x_1 + 40x_2 \quad (4.1)$$

约束条件：

$$0.3x_1 + 0.7x_2 \leq 250, \quad (4.2)$$

$$0.2x_1 + 0.1x_2 \leq 100, \quad (4.3)$$

$$0.2x_1 + 0.5x_2 \leq 150, \quad (4.4)$$

$$x_1 \geq 0, \quad x_2 \geq 0. \quad (4.5)$$

这是一个纯整数规划问题。

解：设工厂每月生产  $A_1$  产品  $x_1$  件， $A_2$  产品  $x_2$  件。则按表 2-1 提供的条件数据， $A_1$  产品  $x_1$  件、 $A_2$  产品  $x_2$  件加工需要的总利润为  $25x_1 + 40x_2$ ，因此，该问题的数学模型为：

$$\max \quad 25x_1 + 40x_2 \quad (4.6)$$

约束条件：

$$0.3x_1 + 0.7x_2 \leq 250, \quad (B_1 \text{ 工序, 工时限制})$$

$$0.2x_1 + 0.1x_2 \leq 100, \quad (B_2 \text{ 工序, 工时限制})$$

$$0.2x_1 + 0.5x_2 \leq 150, \quad (B_3 \text{ 工序, 工时限制})$$

$$x_1 \geq 0, \quad x_2 \geq 0. \quad (\text{且只能整数})$$

另外, 由于  $x_1$  为  $A_1$  的件数, 因此  $x_1 \geq 0$  且只能取整数; 同理,  $x_2 \geq 0$  且只能取整数。

#### Example 4.1.4 ▶ 背包问题（0-1 规划）

**例:** 一个背包的总容积为  $V$ , 现要在  $n$  种物品中选择。设物品  $j$  的重量为  $w_j$ , 体积为  $v_j$ ,  $j = 1, 2, \dots, n$ 。问如何选择, 使得得到的总价值最大, 且总重量不超过  $V$ , 又使装的总重量最大。这一个题目有  $n$  个约束情况, 如装某类装箱, 装箱, 装车等。

**解:** 设对于物品  $j$ , 变量

$$x_j = \begin{cases} 1 & \text{物品 } j \text{ 被装入背包} \\ 0 & \text{物品 } j \text{ 不被装入背包} \end{cases} \quad j = 1, 2, \dots, n$$

则所有被选择装物品的总体积为  $\sum_{j=1}^n v_j x_j$ , 总重量为  $\sum_{j=1}^n w_j x_j$ , 该问题的数学模型为:

$$\max \sum_{j=1}^n w_j x_j \quad (4.7)$$

约束条件:

$$\sum_{j=1}^n v_j x_j \leq V, \quad (4.8)$$

$$x_j = 0, 1, \quad j = 1, 2, \dots, n. \quad (4.9)$$

这是一个 0-1 规划问题。

## 4.2 一般整数规划的求解方法一分枝定界法

为了解决整数规划问题, 我们自然想到两种办法:

- 想到第二章提到的下料问题, 可以用线性规划求解, 结果四舍五入;
- 反正是整数, 不妨穷举求解;

显然第一个方法不够靠谱, 第二个方法效率太低。

**Example 4.2.1 ▶ 第一个方法不靠谱的原因**

例如, 考虑如下整数规划问题:

$$\max 3x_1 + 13x_2 \quad (4.10)$$

$$\text{s.t. } 2x_1 + 9x_2 \leq 40 \quad (4.11)$$

$$11x_1 - 8x_2 \leq 82 \quad (4.12)$$

$$x_1 \geq 0, \quad x_2 \geq 0 \quad \text{且取整数} \quad (4.13)$$

画出可行域如下所示<sup>a</sup>:

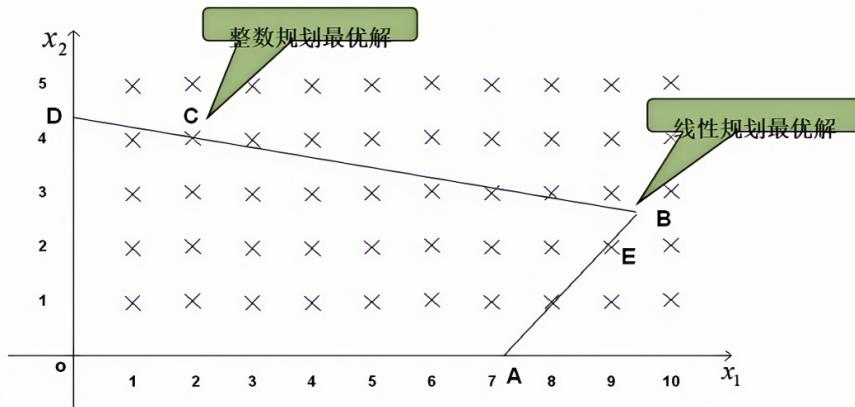


Figure 4.1: 整数规划的四舍五入求解

<sup>a</sup>图中可以看到, 浮点数最优解 B 四周的四个整数点都不在可行域内, 四舍五入的结果是错误的。反而整数最优解在 C 点。

目前, 常用的求解整数规划的方法是分枝定界法和割平面法。作为一种最基本的方法, 下面介绍分枝定界法。

### 分枝定界法

1. 设有最大化的整数规划问题 A，与它相应的线性规划问题(即在整数规划中去掉了决策变量的整数取值要求)为 B，设二者的最优值分别为  $z_A^*$  和  $z_B^*$ ；
2. 从解问题 B 开始，若 B 的最优解符合 A 中的整数条件，则 A 的最优解即为 B 的最优解，结束；
3. 若 B 的最优解不符合 A 的整数条件，则 B 的最优解对应的最优值  $z_B^*$  一定是 A 的最优值  $z_A^*$  的上界，记为  $\bar{z}$ ，而 A 的某一任一可行解(即任意一个整数解)的目标函数值将是一个下界  $\underline{z}$ 。此时  $\underline{z} \leq z_A^* \leq \bar{z}$ ，称为定界；
4. 分枝定界法即不断将 B 的可行域分成子区域(称为分枝)，并在每个子区域中确定 A 的上界  $\bar{z}$  和下界  $\underline{z}$  的方法，逐步夹逼，直到找到 A 的最优解为止；

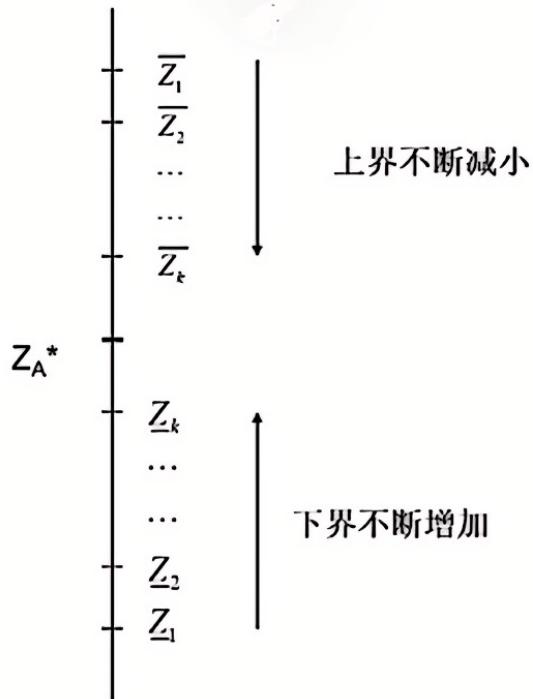


Figure 4.2: 分枝定界法

### Example 4.2.2 ▶ 分枝定界法

例：考虑如下整数规划问题：

$$\max 40x_1 + 90x_2 \quad (4.14)$$

$$\text{s.t. } 9x_1 + 7x_2 \leq 56 \quad (4.15)$$

$$7x_1 + 20x_2 \leq 70 \quad (4.16)$$

$$x_1 \geq 0, \quad x_2 \geq 0 \quad \text{且取整数} \quad (4.17)$$

解：记上述原问题为 A。先考虑 A 中去掉整数约束条件后的线性规划问题 B。

对问题 B 进行第二章中所学到的线性规划分析，可以找到最优解，依照分枝定界法，发现 B 的最优解并非整数，说明需要继续迭代，因此我们记 B 的最优解对应的最优值  $z_B^* = 356$  作为一个上界，并任意找一个 A 的可行解如原点，得到下限  $\underline{z} = 0$ ，此时  $\underline{z} \leq z_A^* \leq \bar{z}$ ，即  $0 \leq z_A^* \leq 356$ 。如下图所示：

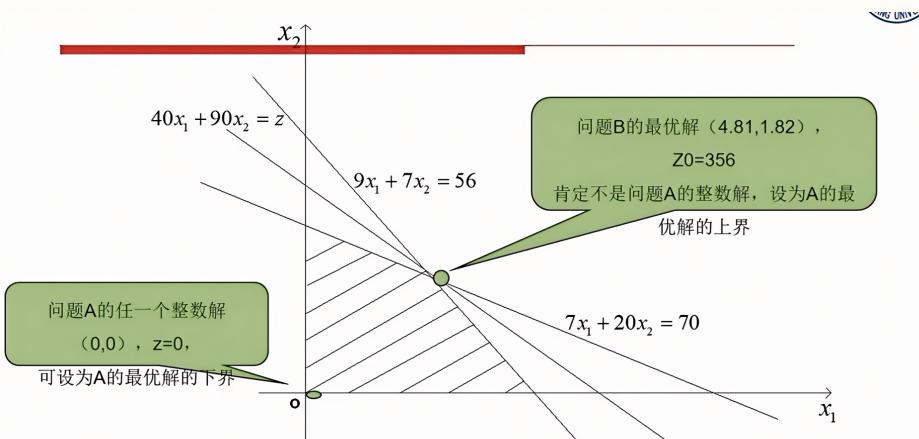


图 2-3 问题 B 的图解分析

$$0 \leq Z^* \leq 356$$

Figure 4.3: 问题 B 的图解分析

在问题 B 中最优解为  $x_1 = 4.81$ ，于是对 B 通过增加两个约束条件  $x_1 \leq 4$  和  $x_1 \geq 5$  可将 B 分解为两个子问题  $B_1$  和  $B_2$ ，即将 B 的可行域在  $x_1 = 4.81$  前后分割为两个子可行域（称为分枝），中间因为不满足整数条件而舍弃，如下图所示：

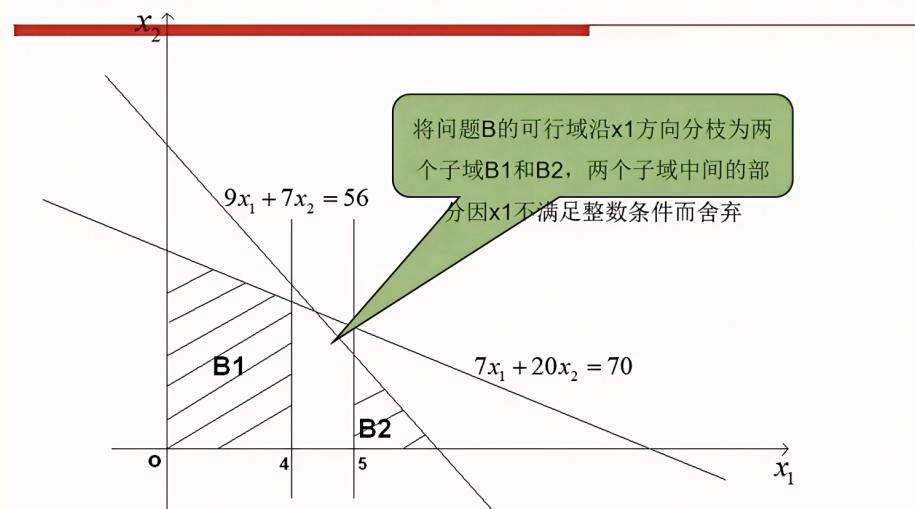


Figure 4.4: 问题 B 的分枝

对应的  $B_1$  和  $B_2$  分别为：

**问题 B1:**

$$\begin{aligned} \max \quad & 40x_1 + 90x_2 \\ \text{s.t.} \quad & 9x_1 + 7x_2 \leq 56 \\ & 7x_1 + 20x_2 \leq 70 \\ & 0 \leq x_1 \leq 4 \\ & x_2 \geq 0 \end{aligned}$$

**问题 B2:**

$$\begin{aligned} \max \quad & 40x_1 + 90x_2 \\ \text{s.t.} \quad & 9x_1 + 7x_2 \leq 56 \\ & 7x_1 + 20x_2 \leq 70 \\ & x_1 \leq 5 \\ & x_2 \geq 0 \end{aligned}$$

这样，我们进行第一次迭代。

在两个区域内分别线性规划求解，如下图所示，两处最优解仍然不都是整数，说明需

要继续迭代。B1 区域的最优值是  $z_{B1}^* = 349$ , B2 区域的最优值是  $z_{B2}^* = 341$ , 取较大的那个, 因此新的上限是  $z_{B1}^* = 349$ , 下限依然是  $z_B^* = 0$ (注意下界此处不是不可以动, 而是我们懒得去动), 所以我们将上限从原来的 356 调整为 349, 此时  $0 \leq z_A^* \leq 349$ 。

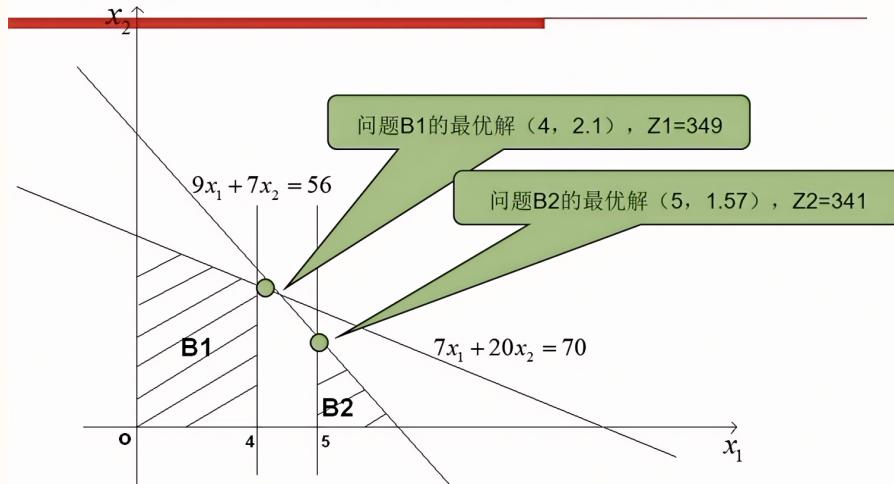


Figure 4.5: 第一次迭代

继续对  $B_1$  和  $B_2$  进行分解。因  $z_1 > z_2$ , 故先分解  $B_1$  为两枝。此次沿  $x_2$  方向分解, 增加约束条件  $x_2 \leq 2$ , 称为问题  $B_3$ ; 增加约束条件  $x_2 \geq 3$ , 称为问题  $B_4$ 。在图 2-4 中舍去  $x_2 > 2$  与  $x_2 < 3$  之间的可行域。

这样, 我们进行第二次迭代。

在两个区域内分别线性规划求解, 可以发现  $B_3$  的最优点仍是顶点, 恰好是整数  $(4, 2)^a$ , 最优值是  $z_{B3}^* = 340$ , 这个数可能是上界(即整数里最优的), 也可能不是, 但至少一定是整数中的一员, 所以肯定可以作为下界<sup>b</sup>。而  $B_4$  的最优值都比不上我们新的下界了, 那  $B_4$  就失去了意义, 不必再分割了; 同理  $B_3$  也不需要再分割了, 最大值是下限, 再求  $B_3$  也没有比 340 更大的整数解了; 但是  $B_2$  有必要继续去做切割。如下图所示:

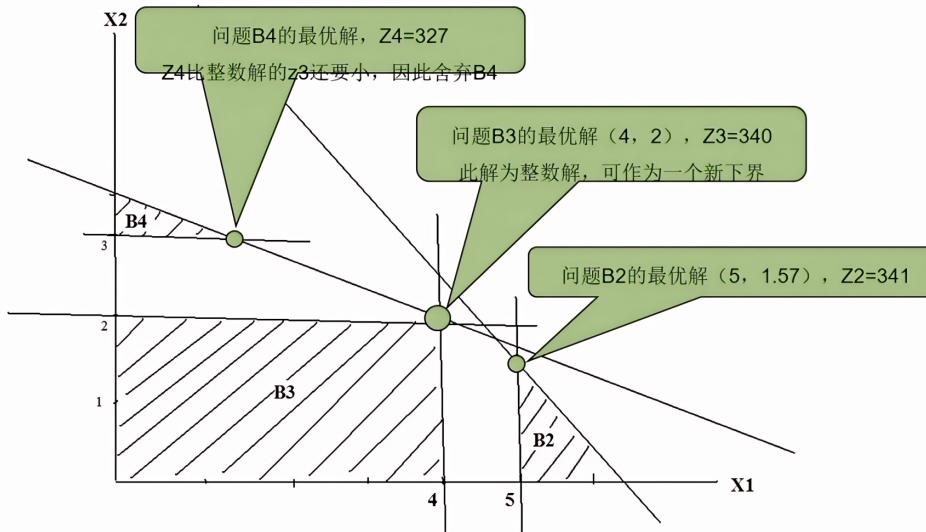


Figure 4.6: 第二次迭代

既然  $B_1$  中已不存在上限，于是分解  $B_2$  得以下问题：

**问题 B5:**

$$\begin{aligned} \max \quad & 40x_1 + 90x_2 \\ \text{s.t.} \quad & 9x_1 + 7x_2 \leq 56 \\ & 7x_1 + 20x_2 \leq 70 \\ & 5 \leq x_1 \\ & 0 \leq x_2 \leq 1 \end{aligned}$$

**问题 B6:**

$$\begin{aligned} \max \quad & 40x_1 + 90x_2 \\ \text{s.t.} \quad & 9x_1 + 7x_2 \leq 56 \\ & 7x_1 + 20x_2 \leq 70 \\ & 5 \leq x_1 \\ & 2 \leq x_2 \end{aligned}$$

这样，我们进行第三次迭代。

在两个区域内分别线性规划求解，可以发现 B6 没有可行解，故 B6 不再需要分割；B5 的最优解  $z_{B5}^* = 308$  小于下限，故 B5 不再需要分割；因此，可以得出结论，最优解就

是  $z_{B3}^* = 340$  如下图所示：

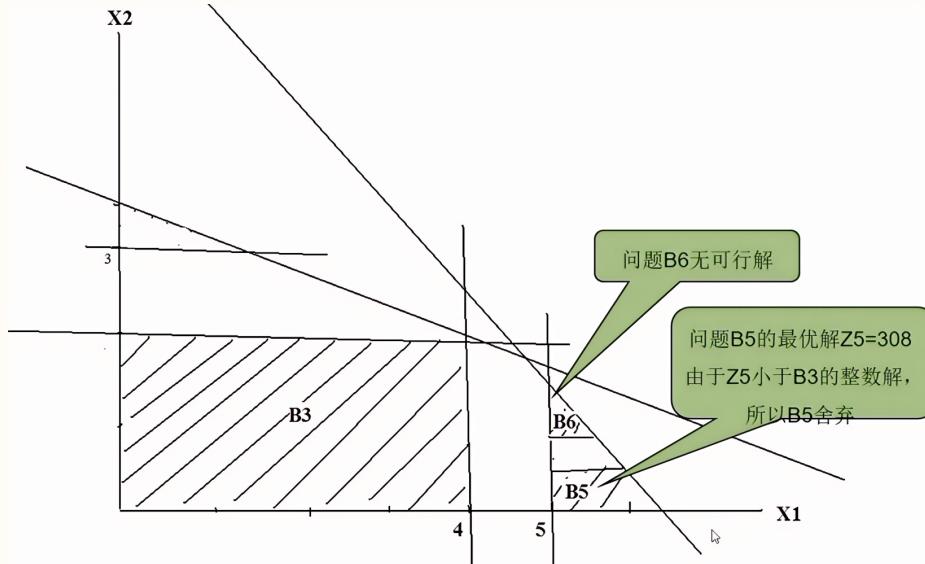


Figure 4.7: 第三次迭代

<sup>a</sup>读者不妨想到此事并非偶然。事实上，由于我们的切割线总是  $x_1 = a$  或  $x_2 = b$ ，总是能割出一个区域为矩形，顶点就是整数解

<sup>b</sup>读者可能会感到疑问，之前方法里不是说算到是整数解就应该停止迭代作为最优解了吗？为什么还要继续算呢？读者不妨想到，这个整数解确实是最优解，但是不是问题 A 的最优解，而是问题 B3 对应的 A3 的最优解，所以仍需要证明其他区域内没有比这个更大或者有更大的最优值了

我们不难发现，在上述过程中，除了最开始找了任意一个整数点作为下限，其他的计算过程全部是线性规划的过程。即过程分为两类：

- (计算过程) 线性规划求解：即求解 B 问题的最优解。
- (判断过程) 分枝：即对 B 问题进行分割，得到子区域 B1、B2、B3、B4 等。同时进行判断并更新上限和下限。

实际上我们在做的，就是先算 B 问题的最优解，如果需要则进行分割，再分别算子区域的最优解。并依据子区域更新上限和下限。实际上是取各个子区域上下限的交集。

- 如果发现子区域最优值小于等于当前下限，则该子区域没有继续分割的必要了。
- 如果发现子区域最优值小于当前上限，则应该更新上限。
- 如果上下限互相逼近到相等，问题可以结束。即证明某一子区域整数最优解所对应的最优值比其他子区域的都要大。

### 分枝定界法的要点

- 分枝: 分为多个更小的空间
- 定界: 确定每个小空间的上、下限和总的上、下界
- 剪枝: 据每个小空间的上、下限和总的上、下界, 删掉那些明显低于总下界和无可行解的小空间。

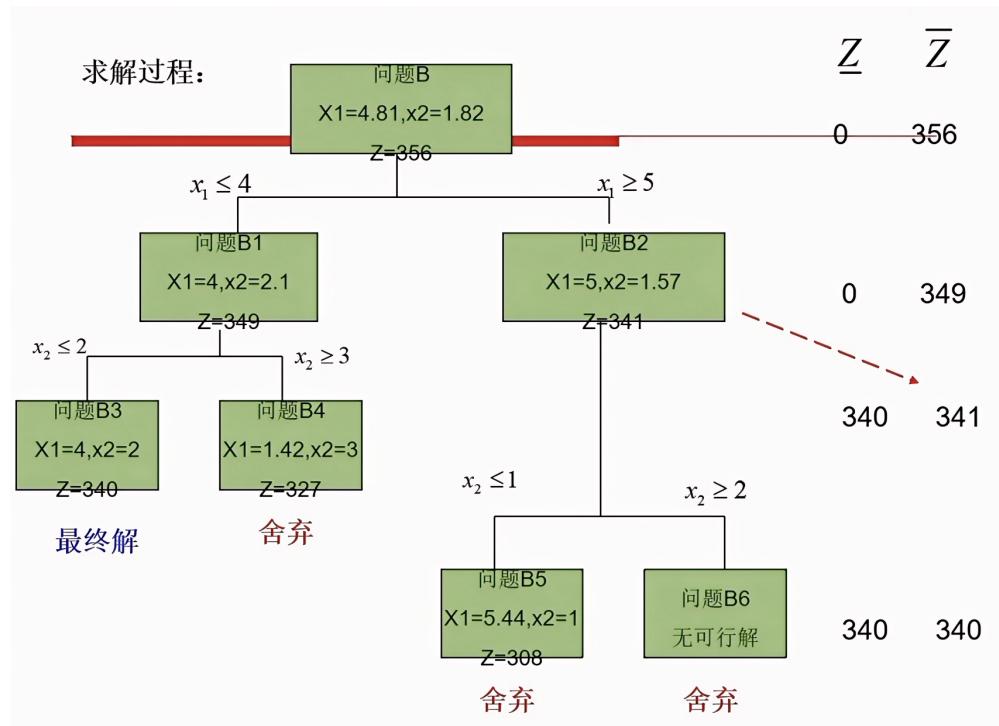


Figure 4.8: 例题总体思路

## 4.3 0-1 规划及其求解方法

### 4.3.1 0-1 规划的定义和数学模型

#### Definition 4.3.1 ▶ 0-1 规划

如果整数规划问题中的所有决策变量  $x_i$ , ( $i = 1, 2, \dots, n$ ) 仅限于取 0 或 1 两个值, 则称此问题为 0-1 整数规划, 简称为 **0-1 规划**, 其变量  $x_i$ , ( $i = 1, 2, \dots, n$ ) 称为 **0-1 变量**, 或**二进制变量**, 相应的决策变量取值的约束为  $x_i = 0$  或  $1$ , 等价于  $x_i \geq 0$  且  $x_i \leq 1$ , 且为整数。

**Definition 4.3.2 ▶ 0-1 混合整数规划**

如果整数规划问题中的一部分决策变量为 0-1 变量，则称为 **0-1 混合整数规划**。

0-1 规划可以是线性的，也可以是非线性的，0-1 整数规划的一般模型为：

**Theorem 4.3.3 ▶ 0-1 整数规划的一般模型**

$$\max \quad (\min) \quad z = \sum_{j=1}^n c_j x_j$$

$$\text{s.t.} \quad \sum_{j=1}^n a_{ij} x_j \leq (\text{或 } =, \geq) b_i \quad (i = 1, 2, \dots, m),$$

$$x_j = 0 \text{ 或 } 1 \quad (j = 1, 2, \dots, n).$$

**4.3.2 0-1 规划的例题****Example 4.3.4 ▶ 销售网点问题**

**例：**某公司拟在城东、城西、城南新建销售网点，有 7 个位置 A1, A2, A3, A4, A5, A6, A7 可选。考虑组建成本和总体布局，规定：

城东：A1, A2, A3 中至多选择 2 个

城西：A4, A5 中至少选择 1 个

城南：A6, A7 至少选择 1 个

若选  $A_i$ ，则建设投资为  $b_i$ ，预计获利  $c_i$ ，但投资总额不能超过  $B$  元。如何选择址获利最大？

**解：**设决策变量  $x_i (i = 1, 2 \dots 7)$

$$x_i = \begin{cases} 1 & A_i \text{ 被选中}, \quad i = 1, 2 \dots 7 \\ 0 & A_i \text{ 未选中} \end{cases}$$

则

$$\begin{aligned}
 \max \quad & z = \sum_{i=1}^7 c_i x_i \\
 \text{s.t.} \quad & \sum_{i=1}^7 b_i x_i \leq B \\
 & x_1 + x_2 + x_3 \geq 2 \\
 & x_4 + x_5 \geq 1 \\
 & x_6 + x_7 \geq 1 \\
 & x_i = 0, 1, \quad i = 1, 2 \dots 7
 \end{aligned}$$

为 0-1 规划。

### Example 4.3.5 ▶ 一般的指派问题（分派问题）

**例：**在实际生产管理中，总希望把有限的资源(人员、资金等)最佳地指派，以发挥其最高的工作效率，创造最大的价值。例如，某科研部门有  $n$  项任务，正好需要  $n$  个人去完成，由于任务的性质和每个人的专长不同，每个人完成各项任务的效率(时间或成本)如表 4.2 所示。如果指派每个人仅能完成一项任务，每项任务仅要一个人去完成，如何分派使完成这  $n$  项任务的总效率(效益量化)为最高，这是典型的标准指派问题。

人 \ 项	1	2	…	$n$
1	$c_{11}$	$c_{12}$	…	$c_{1n}$
2	$c_{21}$	$c_{22}$	…	$c_{2n}$
⋮	⋮	⋮	⋮	⋮
$n$	$c_{n1}$	$c_{n2}$	…	$c_{nn}$

Table 4.2: 指派问题的利润元成表矩阵

**解：**设指派问题的效益矩阵为  $(c_{ij})_{n \times n}$ ，其元素  $c_{ij}$  表示指派第  $i$  个人去完成第  $j$  项任务时所获得的效益 ( $> 0$ )。或者说：以  $c_{ij}$  表示指派给  $i$  的第  $j$  单位资源分配用于第  $j$  项任务时的有关效益。设问题的决策变量为  $x_{ij}$ ，是 0-1 变量，即

$$x_{ij} = \begin{cases} 1, & \text{当指派第 } i \text{ 个人去完成第 } j \text{ 项任务时,} \\ 0, & \text{当下指派第 } i \text{ 个人去完成第 } j \text{ 项任务时.} \end{cases}$$

则其数学模型为

$$\begin{aligned} \max(\min) \quad z &= \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \\ \text{s.t.} \quad \sum_{j=1}^n x_{ij} &= 1 \quad (i = 1, 2, \dots, n), && \text{第 } i \text{ 个人做成一项任务} \\ \sum_{i=1}^n x_{ij} &= 1 \quad (j = 1, 2, \dots, n), && \text{第 } j \text{ 项任务只能由一个人完成} \\ x_{ij} &= 0 \text{ 或 } 1 \quad (i, j = 1, 2, \dots, n). \end{aligned}$$

### 4.3.3 0-1 规划的求解

#### Definition 4.3.6 ▶ 显枚举法

显枚举法(又称为穷举法)是把所有可能的组合情况(共  $2^n$  种组合)列举出后进行比较,找到所需要的解。这种方法对于变量个数较多时,易产生“组合爆炸”,计算量非常巨大。

#### Definition 4.3.7 ▶ 隐枚举法

隐枚举法是从实际出发,从所有可能的组合取值中利用过滤条件排除些不可能是最优解的情况,只需考查一部分的组合就可以得到最优解。因此,隐枚举法又称为部分枚举法。

#### Example 4.3.8 ▶ 隐枚举求解

例: 用隐枚举求解

$$\max \quad z = 3x_1 - 2x_2 + 5x_3$$

$$x_1 + 2x_2 - 3x_3 \leq 2 \quad (1)$$

$$x_1 + 4x_2 + x_3 \leq 4 \quad (2)$$

$$x_1 + x_2 \leq 3 \quad (3)$$

$$4x_2 + x_3 \leq 6 \quad (4)$$

$$x_1, x_2, x_3 = 0, 1 \quad (5)$$

解：对于三变量的 0-1 规划，所有变量组合为

$$(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1).$$

首先，通过试探方法找一个可行解，例如  $(1, 0, 0)$  就是满足所有约束条件的可行解之一，计算  $z(1, 0, 0) = 3$

对于上述极大化 0-1 规划，其最优值  $z^*$  当然大于等于这个初始的任意解  $z(1, 0, 0)$ ，所以有

$$z = 3x_1 - 2x_2 + 5x_3 \geq 3 \quad (6)$$

可以想象，如果以 (6) 来进一步约束原问题 A，则那些  $z < 3$  的变量组合就可以直接舍弃。换句话说，后边只要检查  $z \geq 3$  的变量组合就能确定最优解——这就是“取 0-1 变量组合的一部分”的含义。

然后，把 (6) 标为约束条件，作为一个新的约束条件加入到原问题 A：

$$\max z = 3x_1 - 2x_2 + 5x_3$$

$$x_1 + 2x_2 - 3x_3 \leq 2 \quad (1)$$

$$x_1 + 4x_2 + x_3 \leq 4 \quad (2)$$

$$x_1 + x_2 \leq 3 \quad (3)$$

$$4x_2 + x_3 \leq 6 \quad (4)$$

$$3x_1 - 2x_2 + 5x_3 \geq 3 \quad (6)$$

$$x_1, x_2, x_3 = 0, 1 \quad (5)$$

新问题等价于原问题。构造 0-1 规划约束条件检查表<sup>a</sup>将所有约束条件按 (6)(1)(2)(3)(4) 顺序排好，对每个解依次代入每个约束条件左侧，求出数值，看是否满足约束条件。如果某一约束条件不满足，同一行以下各约束条件就不再检查，因而减少了运算次

数。

整数点组合	约束条件左端计算值 / 右端条件值					满足条件否?	Z值
	(6) $\geq 3$	(1) $\leq 2$	(2) $\leq 4$	(3) $\leq 3$	(4) $\leq 6$		
(000)	0	⊗	⊗	⊗	⊗	✗	
(001)	5	-1	1	0	1	✓	5
(010)	-2	⊗	⊗	⊗	⊗	✗	
(011)	3	1	5	⊗	⊗	✗	
(100)	3	1	1	1	0	✓	3
(101)	8	0	2	1	1	✓	8
(110)	1	⊗	⊗	⊗	⊗	✗	
(111)	6	2	6	⊗	⊗	✗	

取Z值最大的做最优值，最优解为(101)

因为 $6>3$ , 所以继续(1)的计算

条件(6)就不满足, 后边条件不必再检查

因为 $6>4$ 所以不满足条件, 打叉, 后边的条件(3)(4)不必再计算

Figure 4.9: 0-1 规划约束条件检查表

<sup>a</sup>对科幻感兴趣的同學，不妨把这几个约束条件分别视为文明发展中的“大过滤器”，这将有助于理解。对于活到宇宙最后的那些“文明”，再去比比谁发展的好吧！

## 4.4 案例分析

### Example 4.4.1 ▶ 生产计划问题

例：某工厂配套生产某种专业电子产品，今年前 6 个月收到的该产品的订货数量分别为 3000 件，4500 件，3500 件，4000 件，4000 件和 5000 件. 已知该厂的正常生产能力为每月 3000 件, 利用加班生产还可以生产 1500 件. 正常生产的成本为每件 5000 元, 加班生产还要增加 1500 元的成本, 库存成本为每件每月 200 元. 试问该厂如何组织安排生产才能在保证完成生产计划的情况下使生产成本最低? 解：根据这个问题的实际情况, 设  $x_i$  表示第  $i$  个月正常生产的产品数量;  $y_i$  表示第  $i$  个月加班生产的产品数量;  $z_i$  表示第  $i$  个月月初产品的库存数量.  $d_i$  表示第  $i$  个月的需求量 ( $i = 1, 2, \dots, 6$ ), 并且第 1 个月月初的库存为 0, 则问题的生产成本为:

$$\sum_{i=1}^6 (5000x_i + 6500y_i + 200z_i)$$

三项分别为正常生产成本、加班生产成本和库存成本。

### Example 4.4.2 ▶ 游泳队员分配问题

例：某游泳队拟选用 A,B,C,D 四名游泳运动员组成一个 4x100m 混合泳接力队，参加大型运动会，他们的 100m 自由泳，蛙泳，蝶泳，仰泳的成绩如下表 4.3 所示。A,B,C,D 四名运动员各自游什么姿势，才最有可能取得最好成绩？

	自由泳	蝶泳	蛙泳	仰泳
A	56	74	61	63
B	63	69	65	71
C	57	77	63	67
D	55	76	62	62

Table 4.3: 队员的游泳成绩

解：根据题意，假设问题的决策变量为  $x_{ij}$ ，令  $i$  名队员游泳第  $j$  种姿势， $x_{ij}$  的取值如下：

$$x_{ij} = \begin{cases} 1, & \text{让 } i \text{ 名队员游泳第 } j \text{ 种姿势,} \\ 0, & \text{不让 } i \text{ 名队员游泳第 } j \text{ 种姿势.} \end{cases}$$

其中  $i = 1, 2, 3, 4$  分别表示自由泳、蛙泳、蝶泳、仰泳。根据问题的要求可知，四名运动员的成绩矩阵为

$$A = (a_y)_{4 \times 4} = \begin{pmatrix} 56 & 74 & 61 & 63 \\ 63 & 69 & 65 & 71 \\ 57 & 77 & 63 & 67 \\ 55 & 76 & 62 & 62 \end{pmatrix}$$

以  $4 \times 100m$  混合泳所用的总时间最小为目标，以每名运动员游一个项目，每一个项目只能有一名运动员完成为约束，这是一个标准的分派问题。 $4 \times 100m$  混合泳所用的总时间为：

$$T = \sum_{i=1}^4 \sum_{j=1}^4 a_y x_{ij}$$

该模型为一个 0 – 1 整数规划模型。

$$\begin{aligned} \min T &= \sum_{i=1}^4 \sum_{j=1}^4 a_y x_{ij} \\ \text{s.t.} & \quad \left\{ \begin{array}{l} \sum_{i=1}^4 x_{ij} = 1, \quad (j = 1, 2, 3, 4) \\ \sum_{j=1}^4 x_{ij} = 1, \quad (i = 1, 2, 3, 4) \\ x_{ij} = 0 \text{ 或 } 1, \quad (i, j = 1, 2, 3, 4) \end{array} \right. \end{aligned}$$

## 4.5 第四章作业

### 4.5.1 车间生产问题（分枝定界法）

**Q:** 设某服装加工厂有 5 个生产车间，可以用 6 种不同的成品布料（单位为 m）加工不同的服装销售。对于第  $i$  个生产车间分别利用第  $j$  种布料进行加工生产后，可以获得利润为  $r_{ij}$ （元/m） $(i = 1, 2, \dots, 5; j = 1, 2, \dots, 6)$ ，具体的数据表如表 4.5 所示。

该厂现有资金 40 万元，为了分配这些资金，根据各车间的实际生产需求，工厂要求每个车间每种布料至少加工 1000 米，每个车间的总加工能力最多 10000 米，那么试问该工厂每种布料应购买多少米，又如何分配给所属的 5 个车间，使得总利润最大？

布料	加工利润/元					
	1	2	3	4	5	6
车间一	4	3	4	4	5	6
车间二	3	4	5	3	4	5
车间三	5	3	4	5	5	4
车间四	3	3	4	4	6	6
车间五	3	3	3	4	5	7
布料单价/元/米	6	6	7	8	9	10

Table 4.4: 布料单价及加工利润

**A:** 分析如下。

为最大化服装加工厂的总利润，定义以下变量和参数：

- 令  $I = \{1, 2, 3, 4, 5\}$  为车间集合。
- 令  $J = \{1, 2, 3, 4, 5, 6\}$  为布料集合。
- 令  $x_{ij}$  为车间  $i$  分配的布料  $j$  的米数（整数，单位：米）。
- 令  $r_{ij}$  为车间  $i$  使用布料  $j$  的单位利润（元/米），由表 4.4 给出。
- 令  $c_j$  为布料  $j$  的单价（元/米），由表 4.4 给出。

目标函数

最大化总利润：

$$\max z = \sum_{i=1}^5 \sum_{j=1}^6 r_{ij} x_{ij}$$

约束条件

1. 预算约束:

$$\sum_{i=1}^5 \sum_{j=1}^6 c_j x_{ij} \leq 400000$$

2. 最低加工量约束:

$$x_{ij} \geq 1000 \quad \forall i = 1, 2, \dots, 5; \quad \forall j = 1, 2, \dots, 6$$

3. 车间容量约束:

$$\sum_{j=1}^6 x_{ij} \leq 10000 \quad \forall i = 1, 2, \dots, 5$$

4. 整数约束:

$$x_{ij} \in \mathbb{Z}^+, \quad x_{ij} \geq 1000 \quad \forall i = 1, 2, \dots, 5; \quad \forall j = 1, 2, \dots, 6$$

#### Code Snippet 4.5.1 ▶ Matlab 代码

```

1 % 服装加工厂整数规划问题求解
2 clear; clc;
3
4 % 定义参数
5 m = 5; % 车间数
6 n = 6; % 布料种类
7 total_var = m * n; % 总变量数
8
9 % 利润矩阵 r_ij (5x6)
10 R = [4, 3, 4, 4, 5, 6;
11      3, 4, 5, 3, 4, 5;
12      5, 3, 4, 5, 5, 4;
13      3, 3, 4, 4, 6, 6;
14      3, 3, 3, 4, 5, 7];
15
16 % 布料单价 c_j
17 c = [6, 6, 7, 8, 9, 10];
18
19 % 目标函数系数 (最大化转为最小化)
```

```
20 f = -R(:); % 展平为 30x1 向量, 负号因为 intlinprog 最小化
21
22 % 约束矩阵 A 和 b
23 A = zeros(m+1, total_var); % 1 个预算约束 + 5 个车间约束
24 b = zeros(m+1, 1);
25
26 % 预算约束
27 A(1, :) = repmat(c, 1, m); % 每个 x_ij 对应 c_j
28 b(1) = 400000;
29
30 % 车间容量约束
31 for i = 1:m
32 start_idx = (i-1)*n + 1;
33 end_idx = start_idx + n - 1;
34 A(i+1, start_idx:end_idx) = ones(1, n);
35 b(i+1) = 10000;
36 end
37
38 % 上下界
39 lb = 1000 * ones(total_var, 1); % x_ij >= 1000
40 ub = inf(total_var, 1);
41
42 % 整数约束
43 intcon = 1:total_var; % 所有变量为整数
44
45 % 求解
46 [x, fval] = intlinprog(f, intcon, A, b, [], [], lb, ub);
47
48 % 输出结果
49 x = reshape(x, [n, m])'; % 转换为 5x6 矩阵
50 total_profit = -fval; % 最大利润
51 fprintf('最大总利润: %.2f 元\n', total_profit);
52 disp('布料分配方案 (x_ij, 单位: 米):');
53 disp(x);
54
```

```
55 % 每种布料购买量
56 fabric_amount = sum(x, 1);
57 fprintf('每种布料购买量 (米):\n');
58 for j = 1:n
59     fprintf('布料 %d: %.2f 米\n', j, fabric_amount(j));
60 end
```

## 求解方法

采用分枝定界法求解该整数规划问题。由于变量较多（30个），手动分枝定界计算复杂，因此使用 MATLAB 的 `intlinprog` 函数，该函数内置分枝定界算法，可高效求解整数线性规划问题。求解步骤如下：

1. 定义目标函数系数  $f = -[r_{ij}]$ （因 `intlinprog` 最小化，负号转换最大化问题）。
2. 构造约束矩阵  $A$  和右端项  $b$ ，包括预算约束和车间容量约束。
3. 设置下界  $x_{ij} \geq 1000$ ，通过 `intcon` 指定整数约束。
4. 调用 `intlinprog` 求解，得到最优分配  $x_{ij}$  和最大利润。

## MATLAB 代码思路

MATLAB 代码通过以下步骤实现：

- **参数初始化：** 定义车间数（5）、布料种类（6）、利润矩阵  $R$ 、布料单价  $c$ 。
- **目标函数：** 将利润矩阵展平为 30 维向量，添加负号以转换为最小化问题。
- **约束条件：**
  - 预算约束：构造系数向量，限制总成本  $\leq 400000$ 。
  - 车间容量约束：为每个车间构造约束行，限制总加工量  $\leq 10000$ 。
  - 下界约束：设置  $x_{ij} \geq 1000$ 。
- **求解与输出：** 调用 `intlinprog`，输出分配方案、总利润和每种布料购买量。

## 实验结果

运行 MATLAB 代码，得到以下结果：

- **最大总利润：** 234,500 元。

- 布料分配方案 (单位: 米):

$$x_{ij} = \begin{bmatrix} 1000 & 1000 & 5000 & 1000 & 1000 & 1000 \\ 1500 & 1000 & 1000 & 1000 & 1000 & 4500 \\ 1000 & 1000 & 1000 & 1000 & 1000 & 5000 \\ 1000 & 1000 & 1000 & 1000 & 1000 & 5000 \\ 1000 & 5000 & 1000 & 1000 & 1000 & 1000 \end{bmatrix}$$

- 每种布料购买量 (单位: 米):

- 布料 1: 5500 米
- 布料 2: 9000 米
- 布料 3: 9000 米
- 布料 4: 5000 米
- 布料 5: 5000 米
- 布料 6: 16500 米

### 结果验证

- 预算约束: 总成本  $= 5500 \times 6 + 9000 \times 6 + 9000 \times 7 + 5000 \times 8 + 5000 \times 9 + 16500 \times 10 = 400000$  元, 满足约束。
- 最低加工量: 所有  $x_{ij} \geq 1000$ , 满足约束。
- 车间容量: 每个车间总加工量为 10000 米 (如车间 1:  $1000 + 1000 + 5000 + 1000 + 1000 = 10000$ ), 满足约束。
- 利润验证: 手动计算总利润 (如车间 1:  $4 \times 1000 + 3 \times 1000 + 4 \times 5000 + 4 \times 1000 + 5 \times 1000 + 6 \times 1000 = 42000$  元), 总和为 234,500 元, 与输出一致。

### 4.5.2 旅行者背包问题 (0-1 规划)

**Q:** 首先列写模型, 然后自己赋值进行程序求解如下题目: 一个旅行者要在背包里装一些最有用的东西, 但限制最多只能携带  $bkg$  件物品, 每件物品只能是整件携带, 对每件物品都规定了一定的“使用价值”(有用的程度), 如果共有  $n$  件物品, 第  $j$  件物品重  $a_j$  kg, 其价值为  $c_j (j = 1, 2, \dots, n)$ , 问题是: 在携带的物品总重量不超过  $bkg$  的条件下, 携带哪些物品可使总价值最大?

**A: 分析如下。数学模型**

定义以下变量和参数:

- $n$ : 物品总数。
- $b$ : 背包最大承重量。
- $a_j$ : 第  $j$  件物品的重量。
- $c_j$ : 第  $j$  件物品的价值。
- $x_j$ : 二元变量, 表示是否携带第  $j$  件物品 ( $x_j = 1$  表示携带,  $x_j = 0$  表示不携带)。

**目标函数:**

$$\max z = \sum_{j=1}^n c_j x_j$$

**约束条件:**

1. 重量约束:

$$\sum_{j=1}^n a_j x_j \leq b$$

2. 二元约束:

$$x_j \in \{0, 1\} \quad \forall j = 1, 2, \dots, n$$

**具体实例**

我们选择以下数据:

- $n = 5$  (物品总数)。
- $b = 10$  (背包最大承重)。
- 物品的重量和价值如下:

$j$	$a_j$ (kg)	$c_j$
1	2	3
2	3	4
3	4	5
4	5	6
5	6	7

## MATLAB 代码

以下是使用显枚举法求解的 MATLAB 代码：

### Code Snippet 4.5.2 ▶ 显枚举法 Matlab 代码

```
1 % 0-1 背包问题求解（显枚举法）
2 clear; clc;
3
4 % 参数定义
5 n = 5; % 物品总数
6 b = 10; % 背包最大承重
7 a = [2, 3, 4, 5, 6]; % 物品重量
8 c = [3, 4, 5, 6, 7]; % 物品价值
9
10 % 初始化变量
11 max_val = 0; % 最大价值
12 best_comb = []; % 最优组合
13
14 % 枚举所有可能的组合
15 for i = 0:2^n - 1
16     comb = zeros(1,n); % 当前组合
17     for j = 1:n
18         comb(j) = bitget(i,j); % 使用 bitget 获取二进制表示
19     end
20     total_weight = sum(a .* comb); % 计算总重量
21     if total_weight <= b % 如果总重量不超过背包容量
22         total_value = sum(c .* comb); % 计算总价值
23         if total_value > max_val % 更新最大价值和最优组合
24             max_val = total_value;
25             best_comb = comb;
26         end
27     end
28 end
29
30 % 输出结果
31 fprintf('最大价值: %d\n', max_val);
```

```

32 fprintf('选中的物品: ');
33 for j = 1:n
34     if best_comb(j) == 1
35         fprintf('%d ', j);
36     end
37 end
38 fprintf('\n');

```

### Code Snippet 4.5.3 ▶ 动态规划法 Matlab 代码

```

1 % 动态规划法求解
2 dp = zeros(n+1, b+1); % 初始化 dp 表
3 for i = 1:n
4     for w = 0:b
5         idx_w = w + 1; % MATLAB 索引从 1 开始
6         if a(i) > w
7             dp(i+1, idx_w) = dp(i, idx_w); % 不能携带第 i 件物品
8         else
9             take = dp(i, idx_w - a(i)) + c(i); % 携带第 i 件物品
10            not_take = dp(i, idx_w); % 不携带第 i 件物品
11            dp(i+1, idx_w) = max(take, not_take); % 取最大值
12        end
13    end
14 end
15 max_val_dp = dp(n+1, b+1); % 获取最大价值
16 fprintf('动态规划法最大价值: %d\n', max_val_dp);

```

### 求解方法

采用显枚举法求解该 0-1 背包问题。由于物品数量较少 ( $n = 5$ )，共有  $2^5 = 32$  种组合，显枚举法计算复杂度可接受。求解步骤如下：

1. 枚举所有可能的物品组合（使用二进制表示，从 0 到  $2^n - 1$ ）。
2. 对每种组合，计算总重量  $\sum_{j=1}^n a_j x_j$ ，检查是否满足  $\leq b$ 。
3. 若满足重量约束，计算总价值  $\sum_{j=1}^n c_j x_j$ ，更新最大价值和最优组合。
4. 输出最大价值和选中的物品。

为验证结果，采用动态规划法，通过构建二维表格  $dp$ ，计算前  $i$  件物品在容量  $w$  下的最大价值，确保结果正确。

### MATLAB 代码思路

MATLAB 代码通过以下步骤实现：

- **参数初始化：** 定义物品数量 ( $n = 5$ )、背包容量 ( $b = 10$ )、物品重量数组  $a$ 、价值数组  $c$ 。
- **显枚举法：**
  - 使用循环遍历 0 到  $2^n - 1$ ，通过二进制位提取每种组合。
  - 计算每种组合的总重量和总价值，筛选满足重量约束的组合。
  - 记录最大价值和对应的物品组合。
- **动态规划法：**
  - 构建二维数组  $dp$ ，其中  $dp[i][w]$  表示前  $i$  件物品在容量  $w$  下的最大价值。
  - 使用递推公式更新  $dp$ ，比较携带和不携带第  $i$  件物品的价值。
- **求解与输出：** 输出显枚举法的最优解（最大价值和选中的物品），并用动态规划法验证最大价值。

### 实验结果

运行 MATLAB 代码，得到以下结果：

- **最大价值：** 13
- **选中的物品：** 物品 1, 2, 4
- **动态规划验证：** 最大价值为 13，与显枚举法一致。

### 结果验证

- **重量约束：** 选中的物品 1、2、4 的总重量为  $2 + 3 + 5 = 10 \leq 10$ ，满足约束。
- **价值计算：** 总价值为  $3 + 4 + 6 = 13$ ，与输出一致。
- **其他组合验证：**
  - 物品 1、2、3：重量  $2 + 3 + 4 = 9 \leq 10$ ，价值  $3 + 4 + 5 = 12 < 13$ 。
  - 物品 3、5：重量  $4 + 6 = 10 \leq 10$ ，价值  $5 + 7 = 12 < 13$ 。
  - 物品 2、4：重量  $3 + 5 = 8 \leq 10$ ，价值  $4 + 6 = 10 < 13$ 。

- 动态规划验证：动态规划法计算的最大价值为 13，与显枚举法结果一致。

# 非线性规划

## 5.1 非线性规划问题与数学模型

### 5.1.1 非线性规划问题的定义

**Definition 5.1.1 ▶ 非线性规划**

如果目标函数或约束条件方程中存在任何非线性因子，则问题**非线性规划**。

非线性规划具有以下几个特点：

- 目标函数或约束条件方程中存在任何**非线性因子**，或全部为**非线性函数**；
- **没有统一的、通用的解法**(这不同于线性规划，它有单纯形法作为通用解法)，目前各种解法都有自己的适用范围；
- 非线性规划的最优解不像线性规划那样在边界点达到，而是**有可能在可行域中任一点**达到；
- 非线性规划存在**局部极值点**和**全局极值点**之分，这一点也与线性规划不同。

**Definition 5.1.2 ▶ 局部极值点**

设  $x^* \in K$ , 如果存在某个  $\epsilon > 0$ , 使得有与  $x$  距离小于  $\epsilon$  的  $x \in K$  (即  $x \in K \cap \|x - x^*\| < \epsilon$ ), 均满足不等式  $f(x) \geq f(x^*)$ , 则称  $x^*$  为  $f(x)$  在  $K$  上的局部极小点,  $f(x^*)$  为局部极小值。

若  $x \neq x^*$  且  $f(x) > f(x^*)$ , 则  $x^*$  为严格局部极小点,  $f(x^*)$  为严格局部极小值。

**Definition 5.1.3 ▶ 全局极值点**

设  $x^* \in K$ , 如果对于任意  $x \in K$ , 都有  $f(x) \geq f(x^*)$ , 则称  $x^*$  为  $f(x)$  在  $K$  上的全局极小点,  $f(x^*)$  为全局极小值。

若  $x \neq x^*$  且  $f(x) > f(x^*)$ , 则  $x^*$  为严格全局极小点,  $f(x^*)$  为严格全局极小值。

两点说明：

- 局部极值点和全局极值点在定义上的主要区别是点的比较范围是  $x^*$  附近的一个小邻域，还是整个  $K$  域。
- 局部极小点可能同时是全局极小点，全局极小点同时一定是局部极小点。

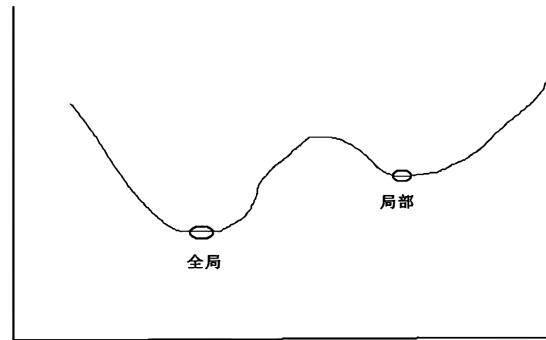
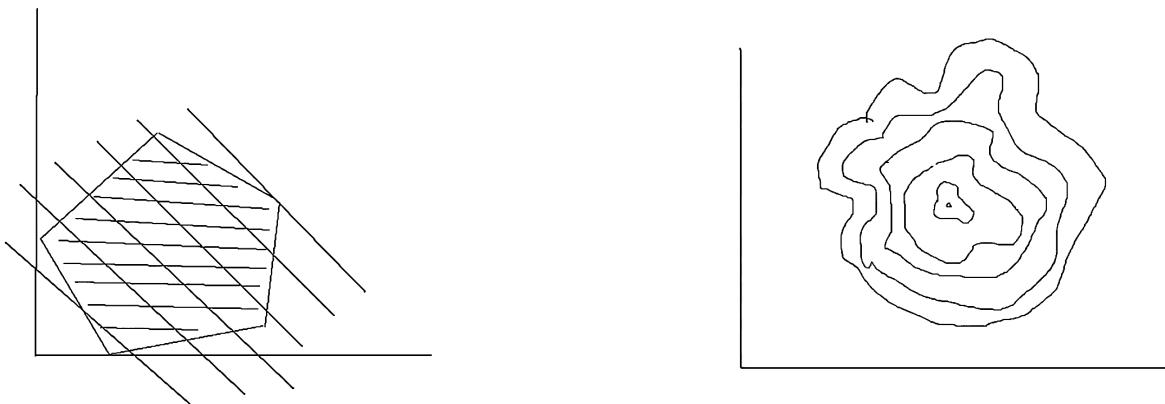


Figure 5.1: 非线性规划也有局部极值和全局极值的区分



(a) 线性规划中，当目标函数取定值时，其等值几何图形为直线或平面。

(b) 非线性规划中，等值线为规则或不规则的曲线、曲面，类似于地图上的等高线

Figure 5.2: 线性规划与非线性规划的对比

#### Example 5.1.4 ▶ 资源分配问题

**例：**有  $A$  和  $B$  两种资源，数量分别为  $a$  和  $b$ ，用于生产  $n$  种产品。如果  $A$  种资源以数量  $x_k$ ， $B$  种资源以数量  $y_k$ ，用于生产第  $k$  种产品，其收益为  $g_k(x_k, y_k)$ ，问如何分配这两种资源用于  $n$  种产品的生产使总收益最大？

**解：**

由题意，以  $x_k$  和  $y_k$  ( $k = 1, 2, \dots, n$ ) 为决策变量，以生产  $n$  种产品的总收益为目标函数，资源的总量为约束条件，则问题的优化模型为：

$$\max z = g_1(x_1, y_1) + g_2(x_2, y_2) + \cdots + g_n(x_n, y_n)$$

约束条件：

$$\begin{cases} x_1 + x_2 + \cdots + x_n = a, \\ y_1 + y_2 + \cdots + y_n = b, \\ x_k \geq 0, \quad y_k \geq 0 \quad (k = 1, 2, \dots, n). \end{cases}$$

### Example 5.1.5 ▶ 最小圆盘问题

例：设平面上有  $m$  个点，找覆盖这  $m$  个点的最小圆盘。

解：设  $m$  个点为  $p_i$ ,  $i = 1, 2, \dots, m$ , 则平面上任一点  $x$  到这  $m$  个点的距离最大者满足

$$f(x) = \max_{1 \leq i \leq m} \|x - p_i\|$$

则以  $x$  为圆心,  $f(x)$  为半径的圆盘必覆盖这  $m$  个点，于是问题转化为求解最小半径的圆盘问题：

$$\min_x \max_{1 \leq i \leq m} \|x - p_i\|$$

这是一个无约束的非线性规划问题。

若没有约束条件，称为无约束极值问题。否则称为约束极值问题。

## 5.1.2 非线性规划问题的数学模型

**Theorem 5.1.6 ▶ 一般形式**

$$\begin{aligned} \max \quad & z = g_1(x_1, y_1) + g_2(x_2, y_2) + \cdots + g_n(x_n, y_n) \\ \text{subject to} \quad & \sum_{k=1}^n x_k = a, \\ & \sum_{k=1}^n y_k = b, \\ & x_k \geq 0, y_k \geq 0 \quad (k = 1, 2, \dots, n). \end{aligned}$$

$$\min \quad \max_{1 \leq l \leq m} \{\|x - p_l\|\}$$

**Theorem 5.1.7 ▶ 标准模型 I**

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to} \quad & h_i(x) = 0, \quad i = 1, 2, \dots, m \quad (\text{m 个等式约束}) \\ & g_j(x) \geq 0, \quad j = 1, 2, \dots, l \quad (\text{l 个不等式约束}) \\ & x \in \mathbb{R}^n \end{aligned}$$

若有另一种形式，可以转化为上述形式，例如：

$$\begin{aligned} \max \quad & f(x) \Rightarrow \min(-f(x)) \\ & g_j(x) \leq 0 \Rightarrow -g_j(x) \geq 0 \end{aligned}$$

**Theorem 5.1.8 ▶ 标准模型 II**

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to} \quad & g_i(x) \geq 0, \quad i = 1, 2, \dots, n \quad (\text{n 个不等式约束}) \\ & x \in \mathbb{R}^n \end{aligned}$$

**模型 I 与模型 II 的转换:**

主要是把等式约束化为不等式约束:

$$h_i(x) = 0 \Rightarrow \begin{cases} h_i(x) \geq 0 \\ h_i(x) \leq 0 \end{cases} \Rightarrow -h_i(x) \geq 0 \Rightarrow g_i(x) \geq 0$$

**求解方法**

求解非线性规划问题常有两种方法:

1. **解析法:** 必要条件 + 充分条件;
2. **数值法:** 迭代。

### 5.1.3 求解非线性规划问题的解析法

为了使用解析方法解决非线性规划问题，我们首先引出几个定义<sup>1</sup>:

**Definition 5.1.9 ▶ 函数的梯度**

设  $x \in K \subset \mathbb{R}^n$ ,  $f(x)$  在  $K$  上有一阶连续偏导数, 则

$$\nabla f(x) \equiv \left[ \frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \dots, \frac{\partial f(x)}{\partial x_n} \right]^T$$

称函数  $f(x)$  在  $x$  处的梯度。

**梯度的几何意义:**  $\nabla f(x)$  是  $f(x)$  在  $x$  处增加最快或减少最快的速度, 也是  $f(x)$  的图形在  $x$  处的陡峭程度。

**Definition 5.1.10 ▶ 平稳点**

若  $\nabla f(x) = 0$ , 则称  $x$  是  $f(x)$  的平稳点。

<sup>1</sup>读者在微积分或数学分析课程中中学过以上概念。如果感到有些遗忘, 可以理解为“梯度”就相当于一元函数的一阶导数, “Hesse 矩阵”就相当于一元函数的二阶导数。我们在判断一元函数的极值点的时候, 就是通过求一阶导为 0, 二阶导大于或小于 0 来判断是极小值或极大值的。对于多元函数也是类似的道理。

**Definition 5.1.11 ▶ 海森 Hesse 矩阵**

设  $f(x)$  在  $K$  上有二阶连续偏导数，则

$$H(x) \equiv \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}$$

称函数  $f(x)$  在  $x$  处的 **Hesse 矩阵**。

**Definition 5.1.12 ▶ 矩阵的正定性，负定性，不定性**

对于对称矩阵  $A$ ，若对任意非零向量  $X \neq 0$ ，二次型为正，即  $X^TAX > 0$ ，则称该二次型为正定二次型， $A$  为正定矩阵<sup>a</sup>。

若  $X^TAX \geq 0 \Rightarrow$  半正定

若  $X^TAX < 0 \Rightarrow$  负定

若  $X^TAX \leq 0 \Rightarrow$  半负定

若既非正定，又非负定，则为不定。

<sup>a</sup>在线性代数中，我们学习过判断正定矩阵的方法有特征值判定法（所有特征值大于 0）、主子式判定法（如果所有阶的顺序主子式，即从左上角开始的子矩阵的行列式都大于 0）；此外，如果两个矩阵都是正定的，那他们的和也是正定的；一个正定矩阵的逆矩阵也是正定的。

**Theorem 5.1.13 ▶ 定理一：极值点必要条件**

若  $f(x)$  在其存在一阶连续偏导数的区域内达到局部极值点，则该极值点必为平稳点。

此处要注意：

- 该定理是必要条件，反之不一定成立，即**平稳点不一定是极值点**，例如鞍点。
- $f(x)$  在其不满足一阶连续偏导数的区域内的极值点，不一定满足定理一。

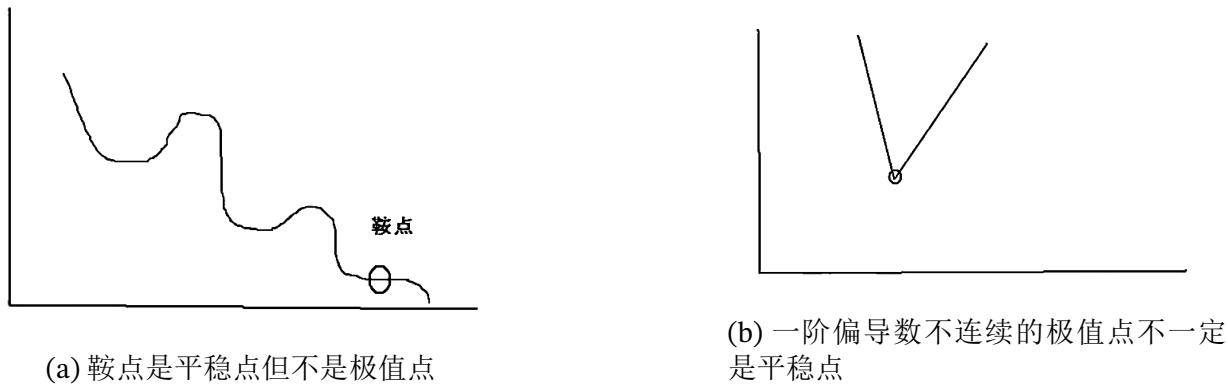


Figure 5.3: 定理一是必要条件的理解

为了能够正确判断极值点，我们使用定理二：

#### Theorem 5.1.14 ▶ 定理二：局部极值点判断的充要条件

若  $f(x)$  在  $K$  上具有二阶连续偏导数，并且满足：

1.  $x^* \in K$  是平稳点，即  $\nabla f(x^*) = 0$
  2.  $H(x^*)$  是正定矩阵（Hesse 矩阵）
- 则  $x^* \in K$  是  $f(x)$  的严格局部极小点。

#### 5.1.4 求解非线性规划问题的数值法

##### Definition 5.1.15 ▶ 下降迭代算法

若由某算法产生的解序列  $\{X^{(k)}\}$  使目标函数值  $f(X^{(k)})$  逐步减小，则称这组算法为下降迭代算法。

##### 下降迭代算法

- 迭代法的基本思路：

初始估计  $X^{(0)}$   $\xrightarrow{\text{按某种算法}}$  比  $X^{(0)}$  更好的  $X^{(1)}$   $\xrightarrow{\text{按算法}}$   $X^{(2)} \dots X^{(n)}$   $\xrightarrow{\text{得到解序列}}$   
 $\{X^{(0)}, X^{(1)}, \dots, X^{(n)}, \dots\}$

若解序列收敛于  $X^*$ ，即，

$$\lim_{k \rightarrow \infty} \|X^{(k)} - X^*\| = 0$$

则称  $\{X^{(0)}, X^{(1)}, \dots, X^{(n)}, \dots\}$  收敛于  $X^*$ 。

几个关键问题是：

1. 迭代算法的初始点  $X^{(0)}$  的选择？
2. 算法的设计，以当前点依据何种原则构建下一个点？
3. 迭代算法的终止条件？

### 迭代法的基本步骤

- **一、初值的确定**

初值的确定没有特别的办法，很多情况下依靠经验，或求解者对问题的了解程度来确定的。初值应尽可能与可能的最优解靠得近一些。

- **二、算法的设计**

1. 选定某一初始点  $X^{(0)}$ ，并令  $k = 0$ 。
2. 确定能使  $f(X)$  下降的搜索方向  $P^{(k)}$ 。
3. 从  $X^{(k)}$  出发，沿方向  $P^{(k)}$  确定迭代步长  $\lambda_k$ 。
4. 确定，产生下一个迭代点  $X^{(k+1)}$ ，

$$X^{(k+1)} = X^{(k)} + \lambda_k P^{(k)}.$$

5. 检查新点  $X^{k+1}$  是否为极小点，或近似极小点，或满足规定的停止条件。
  - 若是，则停止迭代；
  - 否则，令  $k = k + 1$ ，转 (2) 继续进行迭代。

**关键在于如何选取搜索方向  $P^{(k)}$  和步长  $\lambda_k$ <sup>2</sup>。**

方向的确定相对困难，但很重要，避免南辕北辙。

步长的确定可见确定迭代步长的一维搜索方法和确定迭代步长的黄金分割法。

- **三、算法的结束条件**

- 若算法优秀且收敛的极限值确为最优解，则能通过迭代找到最优解。
- 若迭代步数有限，只能找到近似解，当满足所要求精度时，即停止迭代。

常用以下几个方法：

---

<sup>2</sup>各种迭代方法的差异主要就体现在这两者的选定上。

### 1. 两次迭代值的绝对误差

$$\|X^{(k+1)} - X^{(k)}\| < \varepsilon_1$$

$$|f(X^{(k+1)}) - f(X^{(k)})| < \varepsilon_2$$

### 2. 两次迭代值的相对误差

$$\frac{\|X^{(k+1)} - X^{(k)}\|}{\|X^{(k)}\|} < \varepsilon_1$$

$$\frac{|f(X^{(k+1)}) - f(X^{(k)})|}{|f(X^{(k)})|} < \varepsilon_2$$

### 3. 梯度条件

$$\|\nabla f(X^{(k)})\| < \varepsilon$$

## 确定迭代步长的一维搜索方法

### 一维、确定步长的三种方法

1. **恒定步长**: 步长每次不变, 计算简单, 但效果较差。
2. **变步长**: 每次人工调整步长, 效果较好, 但实施麻烦, 且需具备较多的经验。
3. **最速下降步长**: 使沿搜索方向使目标函数值下降最多、最快, 即沿射线  $X = X^{(k)} + \lambda P^{(k)}$  求使目标函数  $f(X)$  的极小,

$$\lambda_k : \min f(X^{(k)} + \lambda P^{(k)}),$$

由于这种方法是以  $\lambda$  为变量的元函数  $f(X^{(k)} + \lambda P^{(k)})$  的极小点  $\lambda_k$ , 故称为一维搜索, 这种确定的步长为**最佳步长**。

以下面这个草图为例, 从零点出发, 方向确定为横轴正方向了, 如果在横轴上走出红色的步长, 纵轴下降红色的部分; 但是如果在横轴走出蓝色的步长, 纵轴下降的部分就会更大。这就是先确定方向, 沿着搜索方向使目标函数值下降最多。

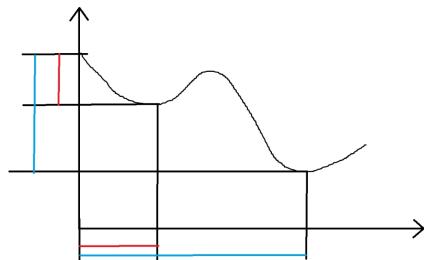


Figure 5.4: 最速下降步长

实际上，读者可能会想到，如果我们每次都要实时计算步长，岂不是降低了效率吗？这个问题是有道理的，请看下面的流程图：

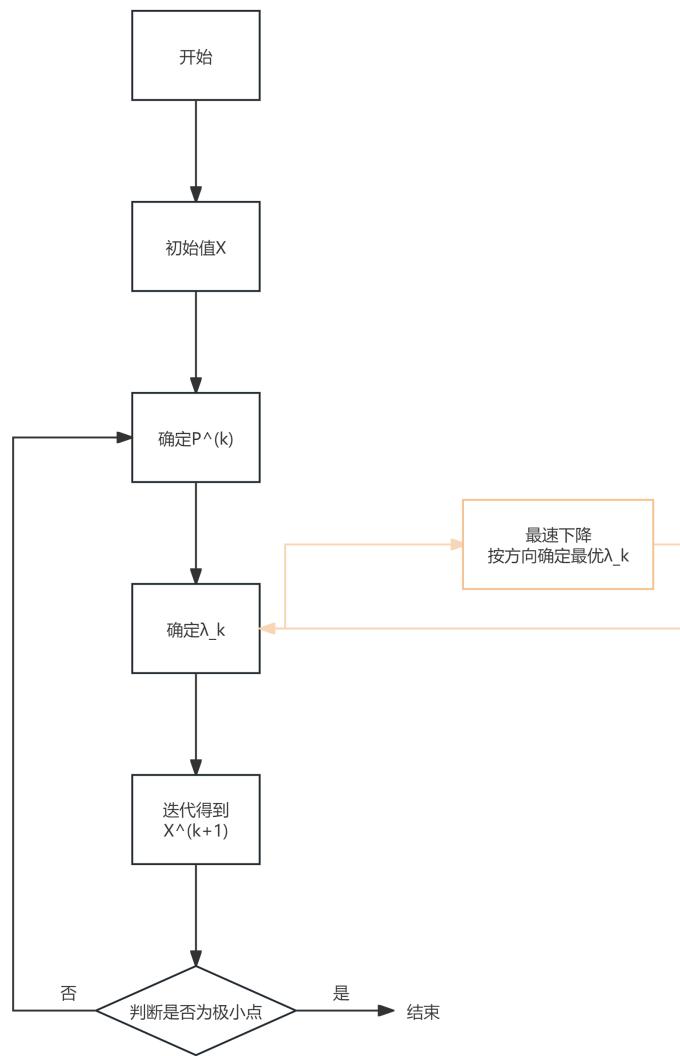


Figure 5.5: 流程图

引入**橙色环节**之前，可能需要迭代 100 次，每次执行 1 秒；引入**橙色环节**之后，可能需要迭代 50 次，每次执行 2 秒。所以总的时间谁多谁少很难说了。因此我们想到，是否有办法在迭代过程中，既能带有最优步长的色彩，计算又不过于复杂？

### 确定迭代步长的黄金分割法（0.618 法）

#### 黄金分割法（0.618 法）

作为一种计算更简便的方法，黄金分割法在区间缩短效率上近似斐波那契法。

$$x'_k = a_{k-1} + 0.382[b_{k-1} - a_{k-1}]$$

$$x''_k = a_{k-1} + 0.618[b_{k-1} - a_{k-1}]$$

其中  $x'_k$  和  $x''_k$  分别是  $[a_{k-1}, b_{k-1}]$  区间内的两个点，0.382 和 0.618 是黄金分割法的两个常数。可以这么理解： $x'_k$  其实是  $a_k$ ， $x''_k$  其实是  $b_k$ ，新的  $\lambda_k$  为  $b_k - a_k$ ，旧的  $\lambda_{k-1}$  为  $a_{k-1} - b_{k-1}$

## 5.2 无约束非线性规划问题的解

## 5.3 约束非线性规划问题（约束极值问题）

## 5.4 案例分析